LoopBack Specifications

Owen Strain
2/9/07
CS190

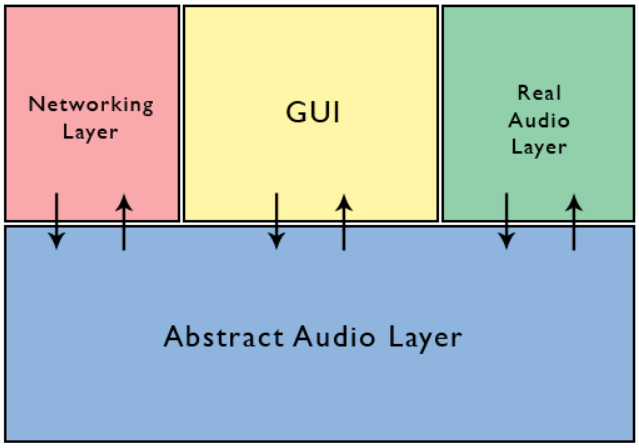
1. Description of Project

A networked music making program that lets multiple users use loops and samples to create songs in real-time.

LoopBack would allow several users to create songs by layering several loops on top of each other. It would use a library of loops so that the only data that must be streamed is metadata about the sound (which loops are playing, volume, speed, timing). No actual audio streaming is needed, greatly reducing bandwidth requirements.

Songs could either be created traditionally by dragging loops onto a timeline and then listening to the result or they could be created in real-time in segments. In the segmented mode, a segment would play in a loop while users add to it and edit it, so that you would hear your changes more or less instantly. When they decide a segment was finished, they could commit it and move to the next segment.

2. System Model

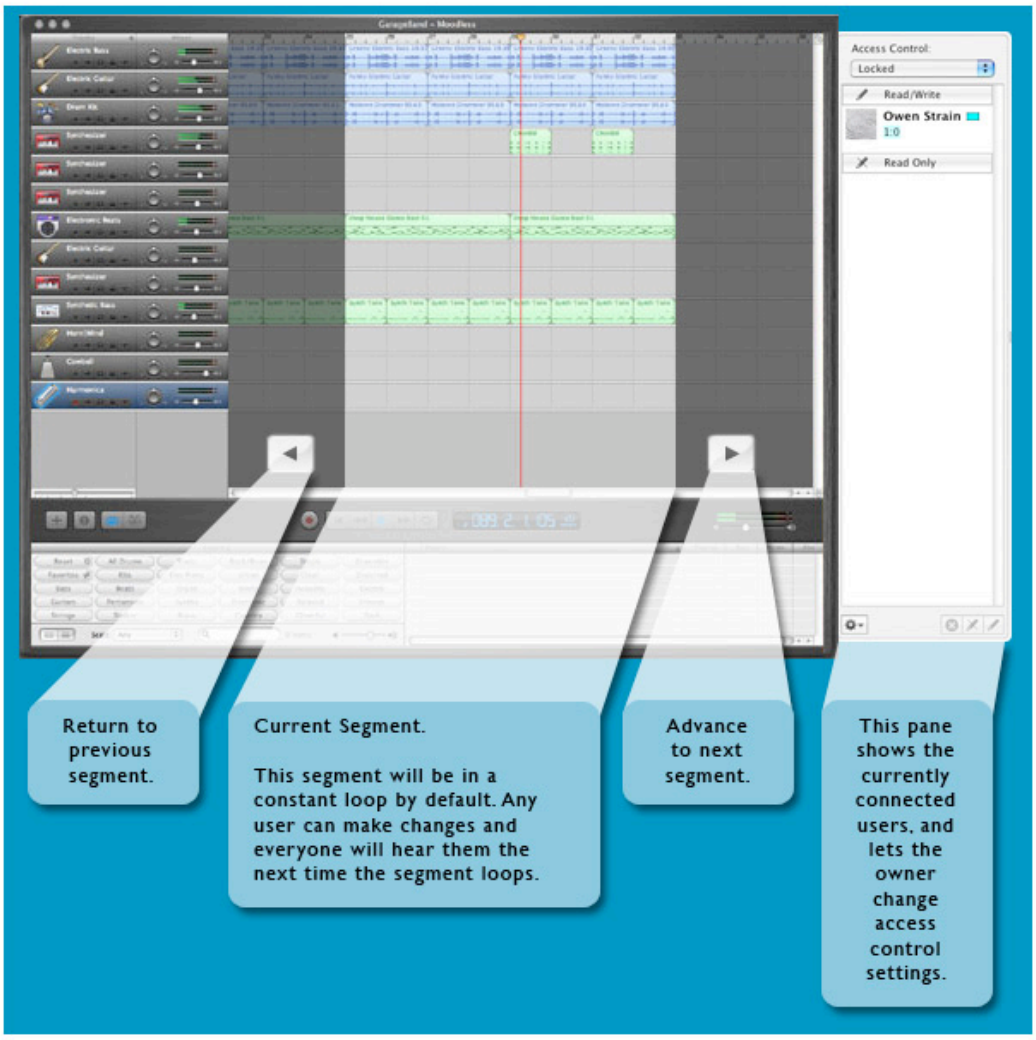


- **Abstract audio layer:** internal representation of the song in terms of loops, timing, speed, and volume information. File format and network protocol would be designed

around this representation. This is the real core of the application — the GUI, networking, and audio processing components all interface with this layer rather than with each other.

- **Real audio layer:** composites together the loops into the actual audio stream. Can play abstract audio from the network or from a saved session.
- **Networking layer:** transmits abstract audio over the network with low-latency (either UDP or TCP). Allows for transmitting incremental changes (e.g. "add loop 5 starting at 74ms playing at 80% speed") and applying them on the remote machine.
- **GUI:** Interface inspired by Apple's GarageBand and similar products. Would be optimized for very fast editing with extensive keyboard shortcuts.

3. User Interface



- Segmented mode (pictured) allows multiple users to edit songs in small segments at a time.
- Participants pane lists all connected users, and gives color coding information to show which users are editing which parts of the segment. For example, if a user's color is yellow and has a certain loop selected, it will be highlighted in yellow for the other users.
- The left-hand side lists all of the tracks, and contains additional per-track controls (volume, etc.)
- The bottom pane contains the library of loops. Users can drag and drop new loops from the library onto tracks in the upper half of the window.

4. Non-functional Requirements

- Performance: real-time audio processing must be good enough to play several tracks without gaps or audible artifacting.
- Ease of use: interface must be tuned for intuitive and rapid editing to allow free-form collaboration between users without the software getting in their way.
- Documentation: the focus should be on making a GUI whose features are discoverable without documentation. Documentation will be needed for non-obvious features like keyboard shortcuts, and for internal components (e.g. the network protocol).
- Portability: should be portable enough to allow ports to Mac OS X or Windows while maintaining file-format and networking compatibility.

5. Requirements

Features

- Create songs using loops and samples
- Intuitive GUI for fast manipulation of loops
- Traditional timeline-based editing mode
- Segmented editing mode
- Networked song creation by multiple users
- Playback of songs and export to WAV / MP3

Advanced / Version 2.0 Features

- Loop editor/creator (MIDI-based)
- Sample editor/creator (MP3-based)
- Online repository of Creative Commons licensed loops that users can upload and share.
- Effects that could be added in real-time (reverb, etc).

- Full multitrack recording / sequencing, ability to combine loops and recorded tracks (e.g. vocals).

Users

- Hobbyists who want to experiment with making songs
- Anyone who has used GarageBand or FL Studio but wants to collaborate on songs with their friends
- One could imagine having song-making competitions between multiple teams, where the teams have a certain amount of time to collaborate on a song over the network

Implementation Roadmap

1. Abstract audio layer
2. Real audio layer (can be started as soon as basics of abstract layer are worked out). Design of this layer will probably affect design of abstract audio layer (same for the networking layer).
3. Networking layer & User interface

6. Challenges / Risks

Networking component

The faster the networking works, the more usable the program will be. Coming up with a protocol to quickly transmit the type of information needed will be a challenge.

UI Design

An intuitive user interface will be very important, because if it takes a user too long to figure out a task, it will slow down the other person/people the user is working on the song with.

Sound

Writing optimized code to combine the various tracks for output will be a challenge, because it must be done in real-time or sound quality will suffer gaps and stutters. This is likely the biggest risk. Hopefully there are libraries that can help with this.