ENSEMBLE: An Object-Oriented Realtime Performance System

Lunette M. Jyer

CCIMA
Stanford University
Stanford, CA 94305
icon@ccima.stanford.edu

Department of Computer Science
California Institute of Technology
Pasadena, CA 91125
icon@parcplace.com

ABSTRACT: The availability of inexpensive realtime synthesizers and powerful, inexpensive personal workstations, has caused computer music systems to rapidly evolve into realtime performance systems that support heterogenous synthesis hardware and a variety of realtime control devices. Ensemble is an object-oriented software system that simulates the dynamic environment of live orchestral performance whereby a conductor coordinates and controls the performance of music by a group of musicians. Ensemble is both an end user application and an extensible software platform for further developing performance capabilities. Ensemble is implemented on a NeXT machine in Objective C and supports both MIDI synthesis and DSP synthesis, as well as MIDI controllers and novel input controllers such as the radio drum.

Introduction

In recent years there has been a growing need for more sophisticated and flexible realtime control systems that take advantage of the power of the new computer technology such as realtime synthesizers and powerful personal workstations. After many years of static taped performances, this new technology has wet the appetite of composers and performers alike for dynamic and expressive performances by machines. Ensemble is an object-oriented software system for coordinating and controlling realtime computer music performance that makes the tremendous potential of the technology more accessible to composers and performers as new devices such as modular synthesis continues to evolve. The system is currently implemented in the configuration shown in Figure 1.

Ensemble addresses the needs of two types of users. The first is a performer/composer who is not a software developer. This type of user "rehearses" a composition by programming the connections between various control inputs and the aspects of the performance that they would like to control, such as tempo, dynamics, etc. The second type of user is also a programmer who would like to enhance the underlying model, add new instruments, etc. For this user, Ensemble acts as an extensible software framework for customizing and experimenting with new control techniques, as well as new synthesis and control devices [Mead].

Once the scores and control patches are rehearsed, Ensemble can perform the composition, allowing a user, using one or more control devices, to dynamically control the performance of music by machines in a live performance.

Basic Model

Ensemble's software model is based on live orchestral performance by a group of musicians coordinated by a conductor. Each live performer has a part of music and the conductor has a score that contains a set of parts. Each performer also has an instrument that is controlled to generate sound and the conductor has a baton that is used to communicate with the performers. The performers read the symbols in their part and map those symbols into...
inputs that are specific to their instrument. The performers also watch the conductor for interpretation information, such as variations in the tempo. This basic model is shown in Figure 2.

In modeling orchestral performance with a computer system, scores and parts printed on paper are replaced with ASCII files stored on magnetic disks. Mechanical instruments that generate acoustic signals are replaced by digital electronic devices that generate electrical signals. The baton can also be replaced with an electronic control device. Performers are modeled by computer software on a personal workstation. This performer software will read a an ASCII file containing the part and, during the performance, compute inputs for their specific synthesizer instrument. Just as human performers develop skills specific to their instrument, the performer software models are customized for specific synthesizers. The conductor is the end user who provides control inputs to the various performer models running on the workstation via a software model of a conductor.

Object-Oriented Design Methodology

Object-oriented programming languages such as Smalltalk-80[Gold83] and Objective C[Cox86] are well-suited for simulating music performance because they provide a powerful abstraction and data modeling capability via classes and behaviors, and a message passing paradigm. Inheritance is also an important feature as it supports a "development by refinement" style to aid in system evolution and code reuse. Ensemble takes advantage of properties of these languages, including dynamic message lookup, late binding, and dynamic unbounded polymorphism to provide the responsiveness and flexibility required for real-time music performance.

Ensemble consists of a set of classes and interface protocols for the basic components of the model, namely scores and parts, instruments, batons, performers, and the conductor. Each of these components has an abstract class and a small set of interface protocols that define the semantics. It is relatively easy for a programmer to customize and enhance the system, as the specificity of particular instruments and performers are implemented by subclasses that override protocols to perform specific tasks.

Figure 1: Computer Music Workstation
Scores

ExeMable is built around a core digital music representation (DMR) that is designed to maximize the realtime control capabilities in the performance (Dyer89). Other file formats such as MIDI files can be used as well. The DMR consists of classes and protocols for manipulating music symbols such as events (notes and rest), event lists (including scores and parts, as well as sequences), note interpretation symbols (such as accent and tenuto), global interpretation symbols (such as tempo and key), and transformations (including, but not limited to transposition, inversion, etc.).

Each event has a duration and an implicit start time expressed in abstract units of beats. It is also possible to specify message sends in the score file. For example, if a note has an accent, the symbol for the accent in the score instructs the system to send the message accent to the performer for the given note. Each performer class may implement a method to do what is required to render an accent on their instrument.

Conductor

A conductor object is used to provide an interface between the control inputs from the user to the software performer models. The conductor class typically has a single instance. The conductor is responsible for maintaining a queue of all of the events in the score and assigning physical start times to each of the performers' events. The physical time is bound at runtime after the previous event is executed. This late binding allows maximum responsiveness to dynamically changing controls, such as tempo. The scheduler sends the performers a message when it is time to play their next event. This is analogous to a human conductor cueing the performers on every note.

The scheduler allows arbitrary messages to be scheduled to provide a high degree of runtime flexibility. The scheduler is designed to accommodate the needs of both static scheduling (when the score is read) and dynamic scheduling (at runtime). Events such as note on events are scheduled dynamically at the time the note is played so that the note can respond to realtime control inputs that effect the articulation of the note.

![Figure 2: Realtime Performance Model](ICMC GLASGOW 1990 PROCEEDINGS 385)
Performers

AbstractPerformer is a class that defines the semantics of performers and concrete subclasses define specific performer types. There is at least one concrete performer class for each synthesis device (MIDIPerformer, DSPPerformer, etc.). More than one is used if the control capabilities of a given type of synthesis device varies between different voices implemented on the device. Initially the performer model can be quite simple but can evolve over time to control increasingly sophisticated aspects of performance.

At the time the score is read from disk, the performer schedules each of the events with the conductor, and may also precompute or cache values to be used at runtime. At runtime the conductor sends each performer the message playEvent at the time each event is to be played. For each performer class, the method computes a set of device specific parameters based on the current realtime inputs and other state information, and in turn sends the message playPacket to the respective instrument. The representation of the packet may vary between instruments, as needed by the underlying hardware and to allow greater runtime responsiveness.

Instruments

The class AbstractInstrument provides the abstraction for a synthesis device. Concrete subclasses are defined for each type of synthesizer used. In cases such as DSP synthesis where a single hardware device implements synthesis models with different semantics, more than one subclass may be used. The concrete class is responsible for implementing the message playPacket, which encapsulates the device driver for the particular synthesis hardware device.

Defining a Performance

A simple user interface is provided by Ensemble to enable the connection of input devices to aspects of the performance. The patching of control data inputs to performance aspects is done in two steps. First, a device independent data value is computed from the incoming (device specific) control data. Then a message is sent with the computed control value as a parameter. The extraction function, as well as the receiver and message, can be programmed by the user at runtime. All extraction functions have a unique name so that they can be configured at runtime. Similarly, all performance objects in the model have a unique name so that they can be specified as the receiver of a message at runtime.

Conclusion

Ensemble is an object-oriented software system that coordinates realtime computer music performance and allows the user to control the performance in a variety of ways. The system is based on a model of live orchestra performance and consists of a set of classes and protocols that model conductors, performers and instruments. A digital music representation is provided for scores and pars that maximizes realtime control capability. In addition to being an end user performance system, Ensemble also acts as a software framework for experimenting with new devices and sophisticated control techniques.

References


ICMC GLASGOW 1990 PROCEEDINGS

386