Sharing Virtual Acoustic Spaces over Interactive TV Programs – Presenting “Virtual Cheering” Application

Tatiana Aires Tavares¹, Aquiles Burlamaqui¹, David André da Silva Albino¹, Clístenes Simonetti¹, Luiz Eduardo C. Leite¹, Jorge Henrique C. Fernandes¹, Guido Lemos de Souza Filho²

¹NatalNet – DIMA – Universidade Federal do Rio Grande do Norte (UFRN)
59072-970 – Natal – RN – Brazil
²LaViD – Departamento de Informática – Universidade Federal da Paraíba (UFPb)
Cidade Universitária - Campus I – João Pessoa – PB – Brazil
{tati, aquiles, davidandre, jaimes, leduardo, jhcf, guido}@natalnet.br

Abstract

This paper presents a proposal to a new class of interactive television programs, which we call immersive television programs. This class of programs promotes group creation to engage in common and cooperative activities in Digital Television platforms, supporting acoustic space sharing. The paper also includes one example of an entertainment program named Virtual Cheering.

1. Introduction

New technological tendencies should provoke radical changes in traditional television next years. The first step is the change in the signal, stopping being analogical and passing to the era of the digital transmission. This way, the paper-key that the television occupies in the society will be increased by the potentialities that the digital era allows.

In the intention of making those services possible through analogical TV sets, it is necessary to use equipments whose basic function is the translation of digital TV signals into analogical ones, the "set-top boxes - STB". STBs are, in the reality, computers dedicated to the task of receiving the digital stream, which is composed of multiplexed video, audio and data elementary streams; to separate the elementary streams; and to send them to the modules responsible for the processing of each one of them. The treatment of the data requests the use of processor, memory, buses and i/o devices. The capacity to transmit and to process data in the digital television systems turned possible the sending of executable code (the named XLets) for the receiving sets: digital televisions or STBs linked to analogical televisions. This way, it is possible to provide to the users a larger control about his programming preferences, and still, to offer other means which expands the concept of Digital TV for Interactive TV [01,02].

The applications that are appearing for the new atmosphere in its majority give support to the Human-Computer interaction. In this article we approached the Human-Human interaction mediated by computer, through the sharing of acoustic spaces. We presented a framework for construction of Interactive Televidio (ITV) applications with that resource type. Although it is necessary to adapt STBs, our proposal opens quite promising perspectives for possible uses in interactive programs in the categories of distributed multimedia applications, more specifically in Entertainment, Music and Distance Learning programs categories.

2. Sharing Virtual Acoustic Spaces

Virtual acoustic spaces designate a relationship among n users of an application where the sound generated by these users is shared by all. To make the acoustic sharing possible, it is necessary to use additional hardware in the STB to capture (microphones) the audio.

Besides the hardware, this service type demands a qualified server to mix different audio streams and to transmit the final result to each user. The result of the mixing is that it supplies to the users the idea of acoustic sharing, because in that way, it is possible to listen to the sound generated by all the other users, as showed in Figure 01. It is also possible to build different levels of acoustic spaces shared in a recursive approach, where the user can choose, for example, to share the space with ten or with hundreds of other users.
3. Proposed Framework

The proposed framework presents a group of classes modeled using UML class diagrams. Our objective is to represent the visions of the user and server side of the application, through classes that translate the main necessary restrictions to support the acoustic sharing.

**ApplicationFrontEnd.** It designates the graphic interface of the application, in other words, the component that it exhibits: the function buttons, legends, and all the other necessary pieces of information. It possesses a reference to an object ApplicationKernel.

**InputControl.** It captures all of the input events generated by the user-viewer. InputControl manipulates and updates ApplicatioKernel. It works as an intermediate between the front-end vision (ApplicationFrontEnd) and the model of the application (ApplicationKernel).

**ApplicationKernel.** This class has the methods responsible for all the communication with the server. It contains information representing the user’s current state (for example, "connected"). This module encapsulates all the functionality of the user.

**ClientThread.** This class is responsible for the reception and sending of control messages to the server (client-server communication channel).

**AudioClient.** This class is responsible for sending the captured audio stream to the server through UDP datagrams; for receiving the mixed audio stream from the server and for storing such a stream in a database.

**Capture.** This class opens an audio line for reading and sends the captured audio to the server through AudioClient.

**Playback.** This class opens an audio line for writing, and play the audio stream stored in the Database.

**DataBase.** This class stores the audio stream that is transmitted from the server to the client and provides controlled access to the stream through the use of an object Traffic signal.

**MessagesPool.** This class implements provides access to the pool of messages received.

**PooledMessage.** This object represents the message used to store the array of bytes containing the audio received from the server.

**Services.** This class represents the other services (chat, survey, etc) available in the ITV application.

**ClientRephaser.** This class is used to make possible the control and synchronization of the audio and video to the users of the TOV client.

The server side of the application is characterized by the classes presented in Figure 03, which are described in the following:

**ApplicationServer.** It is responsible for waiting for users’ connections. It maintains a list of objects ApplicationSlave as well as an object Mixer and a Deliverer.

**ApplicationSlave.** It is responsible for dividing the processing into reception of user messages and reception of control message (Entrance, Exit, Choice of Place, etc.). Each ApplicationSlave possesses an object socket (TCP) related the only one user, an object AudioSlave and a list of received messages (Message).

**AudioSlave.** AudioSlave is the responsible for the reception of audio messages transmitted through UDP datagrams. After having been received, the package is
added to the list of received messages of ApplicationSlave.

Mixer. The mixer is responsible for being constantly consuming the messages of the users' buffers and making the mixing that transforms the several audio streams into only one, which is stored in a list of messages.

Deliverer. It possesses a list of messages already mixed and ready to be transmitted to all of the users, or reviewed for another server.

ServerRephaser. This class is responsible for the control and synchronization of the several users.

The classes of the application were defined in conformance with the ones from the framework proposed. By the user side, a class named TorcidaView presents the attributes of the graphic objects (seats) responsible for the implementation of the virtual physical space, the stadium used as front-end of the application. On the server side, we detached the component Mixer, which is responsible for transforming the several audio flows in only one.

5. Implementation, Experiments and Future Perspectives

The interactive application "Virtual Cheering" was developed in Java, except the video player that uses a library compiled in C++ (player.dll) and accessed through methods native java, JNI (Java Native Interface).

In the Figure 05 we can observe the project of interface of the application that uses the video in full screen and the interfaces of the applications' components are exhibited over the video. We can also observe in that interface solution a concern in maintaining a functional relationship with the available options in the remote control.

The Virtual Cheering application was tested in the following accomplished experiments:

- **Workshop de Redes Avançadas RNP (2002).** The test used video in the MPEG-2 format transmitted at 4 M bits/second rate, the server was located in Natal/RN and a user in Rio de Janeiro/RJ [04].
- **AMPATH Workshop in Miami-EUA (2003).** Test involving a user in Miami, other in Rio de Janeiro/RJ and the server in Natal/RN. A 6 Mbits/second link was used to transmit a 4Mbps MPEG2 video.
- **Simpósio Brasileiro de Redes de Computadores (2003).** The server, located in Natal, users distributed in Natal, Salvador and São Paulo. We accomplished tests of QoS and traffic generated by the application. A 10Mbits/s link was used to transmit a 4Mbps MPEG-2 video.

This application points for the appearance of a new slope of interactive TV programs, that we are
denominating of programs of Immersive TV, once they provoke the sensation of we be submerged in a field twisting together with all the other fans. With the inclusion of 3D elements in this class of programs, the virtual more and more will get confused more and more with the Real turning the immersion census evident.

Our perspectives approach the optimization of the audio distribution using silence detection techniques, to work in the construction of a graphic package of interface components and applications with these facilities and to use the package to implement applications in the distance learning context (to support virtual classrooms) and of entertainment (interactive movies).

6. References


Figure 05. TOV User interfaces: (a) ToV version 2.0 (standard mode and remote control), (b)ToV – Presentation of services, (c) ToV – Chat, (d) ToV – announcement. (e) e (f) ToV – virtual supporter positioning.