YimaCast: A Collaborative Distributed Audio Chat System

1. Research Team

Project Leader:	Prof. Roger Zimmermann, *IMSC and Computer Science*
Other Faculty:	Prof. Wee Ling Wong, *IMSC and Biomedical Engineering*
Graduate Students:	Rahul S. Hampole, Leslie S. Liu, Beomjoo Seo
Undergraduate Students:	Brent Nash
Industrial Partner(s):	Intel

2. Statement of Project Goals

The expanding capabilities of the Internet to handle digital media streams is enabling new applications and transforming existing applications in many areas. One of the fields that is profoundly affected is higher education. Traditionally, students have attended classes in lecture halls on college and university campuses. The next step was distance education, enabled via video satellite links. More recently, new communications media have broadened the potential audience and allowed anybody with a broadband Internet connection to potentially receive audio, video and slide presentations on their computers. The YimaCast project aims to provide a foundation for collaboration tools that would make the student experience more interactive. Our first project is a multi-user audio chatroom system called AudioPeer that is designed to allow groups of students to discuss assignments, allow teaching assistants to conduct lab sessions, and enable student questions and feedback during lectures. Our goal is to provide users with interactive audio experience, which is scalable, practical, integratable and extensible with new features.

3. Project Role in Support of IMSC Strategic Plan

With the YimaCast infrastructure project, we are building a platform that can enable many of IMSC's ongoing basic research achievements into an educational environment, e.g., voice-to-text search, human-machine interfacing, audio concealment, multimedia streaming etc.

4. Discussion of Methodology Used

Our system is designed to provide an efficient audio chat application with low audio latency. To be scalable, we propose a low bandwidth consuming overlay network called *YimaCast*. Moreover, audio mixing is performed at each multicast member to reduce the network bandwidth. YimaCast supports large number of simultaneous audio chat sessions. While YimaCast provides tree management within a session, session-wide management is integrated with existing solutions.

Our system, based on the YimaCast protocol, consists of four components as shown in Figure 1.: AudioPeer, Rendezvous Point (RP) server, Authentication server, and Voice2text index server.

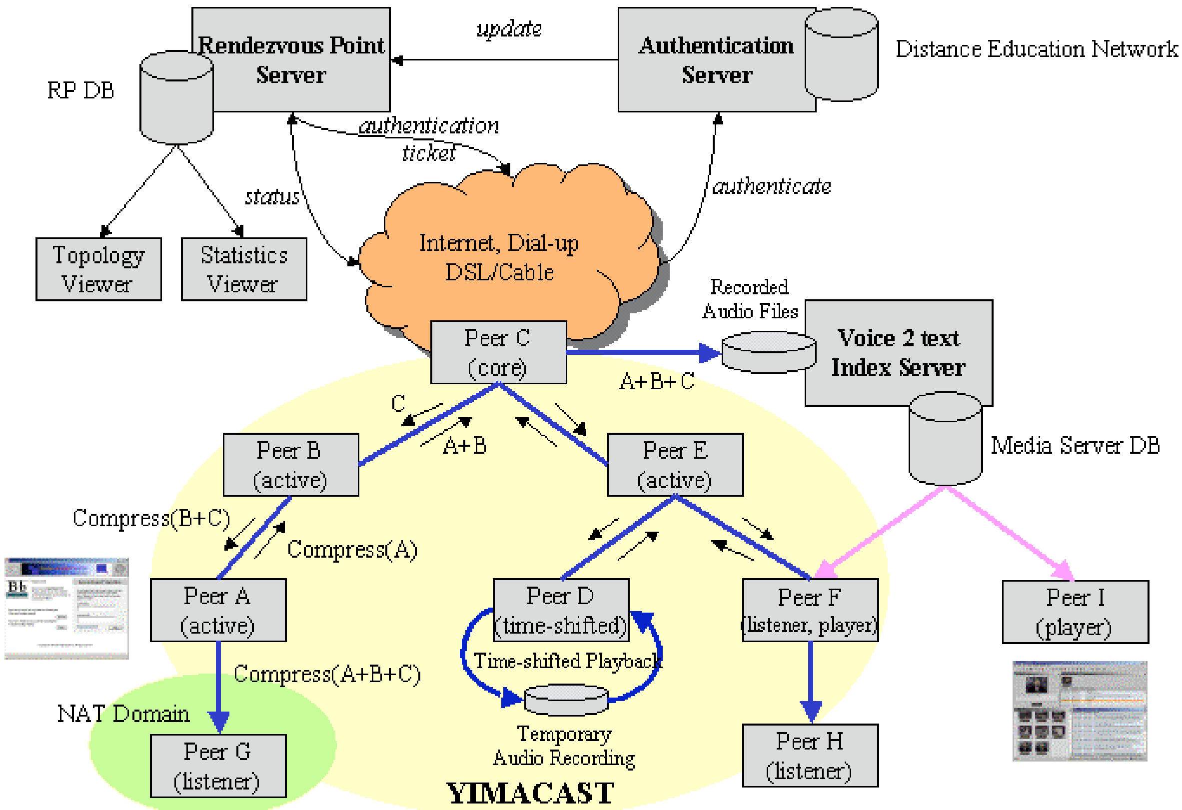

Figure 1: YimaCast is based on a peer-to-peer structure and includes various advanced components such as a topology viewer and a voice-to-text converter.

A brief summary of the techniques employed are as follows:

- Peer-to-peer based architecture to provide good scalability.
- Many-to-many streaming gives virtually no limit to the number of speakers and an enhanced sense of enrollment.
- Software-based audio mixing takes advantage of fast growing computing power (Moore's Law) and enables the system to easily support additional control logic.
- Activity-based dynamic tree rearrangement optimizes the distribution tree to lower the delay between active speakers and listeners while at the same time increasing the audio quality for the listeners.
- The TCP protocol is used to reach users connecting from behind NAT firewalls.
- The Voice2text Index service enables users to index audio materials in text format and perform text-based search on these materials.

5. Short Description of Achievements in Previous Years

All four major components of the system as proposed are currently operational. On December 2, 2003, the final review of the CSCI585 “Database Systems” class was conducted with our system.

5a. Detail of Accomplishments During the Past Year

We implemented the first version of AudioPeer client, which come with a GUI that enables users to configure audio and network parameters. With the embedded web browser, use can join a conference after authentication, do a query on pre-recorded audio files and then retrieve them. All AudioPeer users are organized and cooperate with each other through an application-level multicast protocol called YimaCast. Multiple audio streams can be transported at the same time to all users. Audio streams are mixed at each node to reduce bandwidth requirement. Yimacast can handle unpredictable node failure and automatically reorganize the multicast tree to maintain a system wide consistency. Currently PCM and GSM.610 based audio encoding are supported in our system.

- The Authentication server is built to talk with the existing DEN web service. Students and class information are automatically pulled from the university’s registration database. After the authentication completes, various types of information are provided to help users know current status of the system and decide which session to join.
- We implemented our Rendezvous Point (RP) Server on a Linux server, which has a well know name for all current AudioPeer Clients. RP Server provides the clients with session and active user status. All user activities are also stored in RP Server for research and administration purpose.
- Voice2Text Server is integrated into our system through a dedicated web link shown on AudioPeer Client. Users can perform a keyword search on recorded class sessions and retrieve all relative clips of audio files. Currently the uploading of the recorded audio file onto Voice2Text Server is done manually.

We also conducted an experimental chat for CSCI585, Fall 2003, during which user activity data and comments were collected.

Figure 2: AudioPeer systems integrated with the Blackboard software for authentication running on a desktop and a laptop computer.

6. Other Relevant Work Being Conducted and How this Project is Different

Our system is designed to integrate different components for efficient, easy-to-use, scalable live audio chatroom service: web-based session management, shared tree based application-level multicast delivery, live recording and retrieval, speech recognition service, and audio mixing. There have been several related work done in audio mixing and multicast area. But to the best of our knowledge, no research on providing integrated services like ours has been reported.

Among the existing audio mixing based systems, hierarchical mixing architectures [1] place all participants at the leaf nodes and locate the mixers at non-leaf nodes. A mixer relays the audio coming from its children to the parent after mixing. A root mixer broadcasts the mixed audio to all participants through the IP multicast network. Several commercial products such as the ClickToMeet Conference Product [2], formerly CUseeMe, construct an efficient delivery path (full-mesh or multicast) among the mixers. But because most of these products are integrated with video conferencing applications, they have unnecessary network waste and cannot fully take advantage of audio mixing. In end-system mixing architecture [3], one of the participants works as a mixer. But above systems do not describe how to build efficient delivery paths among mixers and participants.

IP Multicast was originally developed for efficient communication among routers at the IP level by sending a single copy of the transmitted data to all group members. The Multicast backbone (Mbone) was popular in transporting the video and audio through IP Multicast. But unsatisfactory deployment of IP Multicast motivated researchers to suggest various kinds of application-level multicast protocols. Application-level multicast only uses information provided at the application level to transmit data. Various many-to-many protocols are classified as mesh based, tree based, cluster based and Distributed Hash Table (DHT) based architectures. The mesh based architecture connects nodes in a mesh form, e.g. Narada[4]. Each node constructs a single-source multicast tree from the mesh structure. But due to its centralized nature, Narada does not scale well. Tree based architectures include Yoid [5] , HMTP [6] and our proposed YimaCast, which connect nodes in the form of a tree to stream data. NICE [7] was developed as a hierarchical architecture which combines nodes together into clusters and selects parents among these clusters. The process is repeated to form a hierarchy. DHT based architectures use hashing mechanisms to generate node identifiers such that nodes closer to each other logically have similar node identifiers. They then create a multicast tree on top of this substrate. Pastry/Scribe [8] and CAN/CAN [9] Multicast are architectures based on this concept. These systems are designed to support a large number of users with acceptable delay penalty and link stress. All the application-level multicast protocols fail to recognize the feasibility of constructing a more efficient topology if allowing the aggregation of audio packets along the delivery path.

7. Plan for the Next Year

There are several augments we planned for next year. First, we are going to refine our Yimacast protocol, wield it with the capacity to dynamically rearrange the multicast tree based on the user's activity. The basic idea is to put active speakers (users who talk a lot) close to each other on the multicast tree. Also, floor control logic is also being added to Yimacast protocol. Second, we are going to implement an administration interface, which can visualize the on-going chat sessions and provide an operator with various functions to monitor and control the whole system. Third, we will refine our audio encoding module and make it more reliable and reduce the processing delay. Fourth, we will continue to refine the Voice2Text Server and make the processing of recorded audio materials performed automatically by the system.

We are also going to schedule more experimental uses of the system next year. We are planning to do a beta test on a class of Spring 2004 and collect more data. A campus wide deployment of the system on Fall 2004 is also within the sight of our plan.

8. Expected Milestones and Deliverables

Next year we are planning to finish the implementation and test of the new system and give it to DEN (distance education network) for practice use. We are also planning to perform analysis of the data that we collected from DEN after its deployment. Following are some key issues:

- **Refinement of YimaCast Protocol**. New version of YimaCast protocol will includes the capacity to dynamically change the multicast tree to maintain or augment the QoS service in terms of end-to-end delay and audio playback quality.
- **Performance Evaluation.** We'd like to do a large-scale simulation on NS2 platform and compare the performance of YimaCast with existing protocols.
- **Data analysis**. After collecting data through the use of the system at DEN, we are going to conduct some analyze to get the statistics of user activities on the educational chatting environment.

9. Member Company Benefits

The YimaCast infrastructure and the AudioPeer prototype system are being built partially with equipment donated by Intel.

10. References

[1] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. *Lecture Notes in Computer Scienc*e,

[2] C. T. Meet. http://www.fvc.com/.

[3] K. Singh, G. Nair, and H. Schulzrinne. Centralized Conferencing using SIP. In *Internet Telephony Work-sho*p, April 2001.

[4] Y. hua Chu, S. G. Rao, S. Seshan, and H. Zhang. En-abling conferencing applications on the Internet using an overlay multicast architecture. In *ACM SIGCOMM 200*1, San Diego, CA, Aug. 2001. ACM.

[5] P. Francis. Yoid: Your own Internet distribution.

[6] B. Zhang, S. Jamin, and L. Zhang. Host Multicast: A Framework for Delivering Multicast to End Users. In *Proceedings of IEEE Infoco*m, New York, June 23-27, 2002.

[7] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast, 2002.

[8] M. Castro, P. Druschel, A. Kermarrec, and A. Row-stron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in*

communications (JSAC), 2002.
[9] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. *Lecture Notes in Computer Scienc*e, 2233:14–32, 2001.