A Virtual Environment for Collaborative Distance Learning With Video Synchronization

Suhit Gupta
Columbia University
Dept. of Computer Science
New York, NY 10027, US
001-212-939-7184
suhit@cs.columbia.edu

Gail Kaiser
Columbia University
Dept. of Computer Science
New York, NY 10027, US
001-212-939-7000
kaiser@cs.columbia.edu

Abstract

We present a 3D collaborative virtual environment, CHIME, in which geographically dispersed students can meet together in study groups or to work on team projects. Conventional educational materials from heterogeneous backend data sources are reflected in the virtual world through an automated metadata extraction and projection process that structurally organizes container materials into rooms and interconnecting doors, with atomic objects within containers depicted as furnishings and decorations. A novel in-world authoring tool makes it easy for instructors to design environments, with additional in-world modification afforded to the students themselves, in both cases without programming. Specialized educational services can also be added to virtual environments via programmed plugins. We present an example plugin that supports synchronized viewing of lecture videos by groups of students with widely varying bandwidths.

Key Words

Collaborative environments, metadata, visualization, video synchronization

1. Introduction

Learning is in large part a social activity [22]. Enabling social interaction among students is of paramount importance in courses based on team projects, where a high degree of cooperation is required, but challenging to achieve among distance learners. There has been research in the field of collaborative environments for seamless association between non-collocated users. Approaches have ranged from asynchronous systems like email and synchronous applications like instant messaging to more sophisticated approaches like collaborative virtual environments (CVEs) where users can visualize their teammates in a MUD (multi-user domain) environment.

CHIME (Columbia Hypermedia IMmersion Environment) is a CVE that supports *groupspaces* [12], enabling, users to interact with one another through avatars and manipulating objects in the world that are mapped to user-specified backend data sources. Instructors set up the collaborative information space by supplying CHIME with references to data sources they wish to visualize. The server then extracts metadata from the specified source (*metadata* is just data about data [6]). It is this metadata that is ultimately visualized, after 3D model information is applied to it based on either preset defaults or instructor-specified visualizations. The world itself serves quite effectively as an authoring tool, can be viewed and manipulated by teams of users, and can be augmented at runtime allowing users to pull in related educational information as needed.

CHIME includes a general plugin framework supporting special tools, such as remote laboratory resources, but to date only one substantial example has been implemented. This plugin, VECTORS (Video Enhanced Collaboration for Team Oriented Remote Synchronization), enables a group of distance learning students to watch lecture videos in synchrony, seeing “the same thing at the same time” *semantically*, even though users with different bandwidths may see different frame sequences. This plugin framework intends to address similar goals to MIT iLabs (http://i-lab.mit.edu), which focuses on web-accessible labs for science and engineering courses.

The above models helped create a framework well-suited for distance education. We present CHIME’s user view, followed by the architectural components that enable our in-world authoring mechanisms, and describe two generations of the video synchronization facility. We discuss related work and conclude with future directions. Due to space constraints, all figures are at the end.

2. CHIME User View

CHIME logs the user onto the system (Figure 1) by taking the user’s access credentials from the client and forwarding to the server. The user finds herself in an empty starting room from where she can choose to enter a pre-built space (one or more rooms) by opening one of the existing doors, or start building a new space.

Once a space is populated, objects in the world can be clicked on, moved to different locations, or right-clicked to access more information. One can move objects from one room to another without affecting the backend data sources. This is highly advantageous, as an instructor could set up the initial space for a class, choosing a variety of educational materials from many heterogeneous data sources, but reflected according to a "theme" in one homogenous environment. Students can then rearrange them as desired (as permitted by access controls). Their changes are reflected to all users.

Additionally, each user has her own 2.5D overview map of the world to help with navigation. To enhance the group experience, instant messaging and chat are built in. Each user is also provided with a history of her actions, to make it easy to navigate back to a previous room. Example screenshots can be seen in Figures 2, 3 and 4.

3. CHIME Architecture and Components

CHIME is a framework for distributed metadata-based information management and visualization environments, designed to enable sharing of heterogeneous applications and data in a homogeneous virtual world. Our key insight is to not directly map the actual data into the world, which typically would involve substantial programming, but instead to automatically extract metadata about the data specified and visualize that metadata according to pre-programmed 3D objects. This also saves the system from downloading large amounts of data *a priori* and/or mirroring it in a local repository. The actual data is retrieved over the Internet only when a user actually accesses it, and then shown separately in the appropriate data-specific editor or viewer (as in a Web browser).

FRAX – CHIME is composed of a centralized server component and many distributed 3D client front-ends. When data is selected to be added to a virtual world, the CHIME server invokes its metadata extraction engine, *FRAX* (File Recognize And XMLify). FRAX is a general purpose metadata extraction and management toolset and API. While it was engineered primarily to satisfy CHIME's requirements, it can also serve as a standalone component. FRAX addresses two key challenges: 1. the extraction of relevant metadata from an extensible set of data sources; and 2. the efficient management and caching of relevant metadata. Analogous to a Web browser, FRAX attempts to communicate with the indicated data source using one of many preconfigured protocols (HTTP, FTP, SQL, etc.). Once the connection has been established, it uses MIME type matching to determine the kind of component that needs to be used in order to extract metadata. FRAX then uses one of several pre-written components to extract metadata from the data source. Further FRAX components can be written by extending the default interface provided with the system. The metadata extracted by the component is represented as an XML document to capture structural relationships and provide a rich source of information to the server.

VEM - The extracted metadata is parsed and placed in a database where 3D model information (stored in 3DStudio's .3DS format) is assigned by the server's Virtual Environment Modeler (*VEM*), based on an administrator's or instructor's provided preset defaults. VEM's 3D model repository can be extended by a graphics artist. The assignment of the 3D representation can be done via simple mapping rules or according to complex "themes" (visually related collections of 3D objects). Container relationships or hypermedia links that may exist in the data can be followed, upon author request, and are represented as doors. The new space created beyond the door is populated with data representing the containee(s) or followed link. Therefore, instructors (or TAs) can quickly and easily set up the environment they want for their students.

The updated contents of the virtual world are then pushed to currently connected clients in the affected area, if any; the virtual world can thus be modified while in use. This is accomplished using an Internet-scale, publish-subscribe event system, U. Colorado's Siena [7].

4. In-World Authoring

An author identifies each data source to include in the world with a URI-like reference and user credentials, for FRAX metadata extraction. The author then specifies the VEM mapping between classes of metadata objects and 3D models. The client displays the appropriate 3D objects, typically according to built-in layout algorithms. However, the author of a space can provide specific layout information for each of the objects, and can also incrementally update object class-to-3D model mappings as desired. All this is done while the authoring user is "in" the virtual world. If a space is modified while other clients are in the affected room(s), the scene is redrawn for them (after prompting to avoid disorientation).

An instructor would typically create the world and populate it with class materials, such as the course website and documents required for completion of assignments and projects. Consider container type objects, e.g., representing directories in the backend source. When the instructor double-clicks within CHIME, the directory is automatically reflected in a new room extending from the previous room, and populated with all the objects mapped from those directories. Thus, a hierarchy in the virtual environment mirrors (the metadata of) the original data source. An instructor can also define files to be containers of application-specific attributes. For instance, an HTML file can contain links, images and text; a Word document could be comprised of images, tables, text and equations. Some data, of course, are represented as atomic objects, either furnishings or decorations in rooms. Given a FRAX component for extracting metadata from each (sub) element as well as VEM 3D objects with which to represent them, CHIME can visualize any data type. Since a CVE does not have to match real world geometry, one can have an arbitrarily complex world layout – although simple is recommended.

The above process allows student users as well as instructors to create rooms, add new objects inside rooms, etc., to the degree permitted by the access controls defined by the instructor (or administrator). User changes are seen by other users in the same space.

5. Lecture Video Synchronization

One of our goals was to integrate video synchronization for groups of students. The Columbia Video Network (CVN) offers taped courses over the Internet, primarily to MS candidates in the engineering school. These courses work well when the class is simply a series of lectures and homework assignments that do not require group work. However, for courses like Software Engineering, where a large team project is one of the pedagogical requirements, CVN is unable to deliver an experience comparable to that of on-campus students, since the students registered for CVN courses are geographically dispersed and often never meet each other in person. Further, CVN students can benefit from study groups for non-project courses, where they might want to watch lecture videos together and discuss them while in progress (or "paused").

Students are not required by CVN to have the same bandwidth or computation resources. To facilitate synchronized video feeds to diverse clients, we deliver pre-processed semantically structured videos over the heterogeneous Internet links to heterogeneous platforms in an efficient and adaptive manner. The semantic structuring of the CVN lecture videos is done by an algorithm developed by Kender and Liu [17], who pre-process the taped lectures and extract a series of jpeg images that are representative of the entire video segment. Instead of following approaches like those employed in commercial multimedia applications like Real Player or QuickTime that drop every nth frame upon encountering network lag, which may have the negative side-effect of dropping important segments of the video, their process produces several levels of key frame density, each feed targeted at a different bandwidth level. Their algorithms are optimized for typical lecture videos, with a particular emphasis on capturing unobstructed views of full blackboards. (They are not intended for rapidly changing action such as sports clips.) Given the resulting set of video streams, each stream consisting of the best representation of the content for a particular bandwidth level, our goal was to give each user the best possible set of frames while staying synchronized with other users simultaneously watching the same video.

Our approach is three-fold:
1. Prefetch as many of the key frames as possible at the highest possible quality to each client before a pre-determined meeting time for the group. However, videos may be watched immediately without any prior notice.
2. Probe the clients' bandwidth as well as their video cache, and report these results periodically.
3. React to bandwidth changes in real time by lowering or raising the client to a lower or higher quality feed.

All the video stream feeds are made available by the video server (distinct from the main CHIME server). Probing is done by using software probes [14] [15], with reports of any changes sent to the respective clients. Based on which video frames it has in cache, its current position in the video and its current bandwidth, the client determines the highest quality frame it can retrieve in the time remaining before it must be viewed, and downloads it. This continues until the end of the video. If a client finds itself lagging, it automatically drops to a lower quality stream in order to catch up, and will move to a higher resolution feed when possible. See Figures 5 and 6.

We used a testbed of up to 10 clients with clock speeds ranging from 400MHz to 3GHz, and connection speeds ranging from 56Kbit modems up to 100Mbit Ethernet. The videos always synchronized between all 10 clients within an error of 4.38 seconds, i.e., at no point was any client-viewed frame more than 4.38 seconds ahead or behind any other. However, after approx. 7 minutes, independent of which video was playing, the testbed started showing more of a disparity on the laptops without native 3D hardware support built in – which therefore have to render the virtual environment in Software mode.

The problem appears to be caused by a combination of inefficiencies in the 3D graphics engine (CHIME uses the open-source Crystal Space, http://crystal.sourceforge.net), and the fact that the video synchronization uses the same peer-to-peer UDP streams among clients that are also used to track user movements in the virtual world. The server knows where all the users are at any given time, but only at a room level granularity. As mentioned earlier, all communication between the server and the clients takes place over the event system. However, since user position synchronization is a high frequency process, the publish/subscribe system did not make for a good vehicle for this job, especially since the event system would add a substantial filtering and routing latency to each event even if it was as simple as coordinates in 3-space. The server instead sends every client an updated list of users (clients) in the same room and the client then sends position updates to each of these other clients over a UDP stream. This peer-to-peer model is a proven one that works well for commercial massively multiplayer game systems [13], but broke down with the video synchronization overload

Our lab then developed another video synchronization system, AI^{2}TV (Adaptive Interactive Internet Team Video) [21], which operates independently of the 3D world in another window on the user's desktop. It uses the same video server with the same semantically compressed videos. Here an "autonomic" feedback control loop monitors the clients and dynamically adjusts their configurations. AI^{2}TV inserts sensors [14] into the video clients to determine what frames are actually being shown, and the actual bandwidth. All sensor data is input to a workflow engine that instantiates and coordinates local actuators to dynamically adjust for each client the selection of which compression level and which next frame to pull from the video server. The workflow also instructs clients to prefetch from possibly higher resolution streams into their caches during idle time, e.g., when the video is "paused". Experimental trials showed much higher "goodness", our metric weighting aggregate resolution and skew. The next step is to integrate AI^{2}TV into CHIME, to approximate the view of Figures 5 and 6.

6. Related Work

One key concern of research in educational technology has been social interactions, e.g., attempts to improve the user interfaces to enable seamless communication with others [2] [22]. Such research has focused on identifying potential indicators of effective collaboration and the types of problems that may result from insufficient group interaction and support [8]. Prasolova-Førland discusses the mechanisms employed to improve social awareness in education [22] [23] and has found that traditional collaborative technical tools like ICQ and email are not enough, and the mechanisms offered by CVEs provide a promising supplement to the mechanisms in use already.

Brouras *et al.* [3] [4] describe a robust CVE that supports education. However, their environment can only be modified by editing VRML files whenever new materials need to be introduced. This requires technical expertise that a typical instructor may not possess, and changes cannot be made to the world while in use. Okada *et al.* [19] present a system that supports setting up a CVE for ecological education. While users can add virtual areas at runtime, the setup process is tedious: The user is responsible for uploading all the data about the environment she wishes to create as well as giving layout information to the server. CHIME overcomes this by doing all the data extraction and assembly for the user.

Oliveira *et al.* [20] bring the idea of CVE-based collaborative learning to industrial training and e-commerce. Their environment supports video on demand, but without synchronized video playback - so trainees can only discuss videos they have watched separately. J. Liu *et al.* [16] describe a system similar to our video synchronization plugin. However, they are primarily concerned with the QoS of the video, and therefore their approach involves compression techniques working with Mpeg-7 video. They do not address embedding their video stream in a CVE or a collaborative tool of any sort.

An earlier version of CHIME followed a largely different architecture and was designed to support distributed software development environments [11] [12]. The users then were software project team members, possibly geographically dispersed, but *virtually* collocated within the same "room" or adjoining "rooms" of a 3D world. The layout and contents of such a groupspace represented the software project artifacts and/or the on-going software process. Doppke *et al*. investigated a similar idea [10], but with a text-based user interface.

Kaplan *et al.* [18] describe their Orbit system, which supports a groupspace model - although not a CVE and without data or metadata visualization. Orbit best provides access to data that is physically located on its server. While they do allow references to external data, this is limited to a Web link without ability to incorporate what is at the end of the link. CHIME supports multiple protocols over which to connect to backend data sources.

Finally, a significant advantage of 3D CVEs over other kinds of collaborative environments is the ability for users to simultaneously "see" what several of their peers are doing in the groupspace. Remote desktop sharing does not scale to watching different users doing different things with different data. Compared to other CVEs, educational or otherwise, the CHIME approach provides a seamless in-world, real-time authoring environment readily accessible to non-programming instructors and students – while also supporting programming of advanced services such as group-synchronized lecture videos.

7. Conclusions and Future Work

We have presented a 3D collaborative virtual environment for use in distance learning education. External on-line materials are analyzed upon a user's request and corresponding metadata is automatically reflected in the layout and contents of the virtual world at runtime, without programming. Users view and manipulate the internal contents of such materials with the usual local application programs, as in a Web browser. CHIME also enables groups of users to view lecture videos synchronously, with "pause" and other VCR functions, over diverse and fluctuating bandwidths.

There are, however, some limitations. The 3D environment is extremely resource intensive with respect to the CPU and graphics card. This is mainly due to our use of dynamic rather than static 3D objects in the Crystal Space 3D engine. We aim to work with the developers of the engine to help alleviate some of the loads, although many prospective laptop users will likely update to new laptops with 3D chipsets. We also need to improve the way users view the state of their peers in the environment, e.g., to show a graphical depiction of avatars actually holding or manipulating the objects those users are working with. At present selecting the user indicates in text what she is doing, and selecting an object states in text which user if any is using it. Finally, benchmark tests have shown that the Siena event system, used for CHIME's client/server updates, can process only about 400 events per second per event router. We are converting to the Elvin event system (http://elvin.dstc.edu.au), a higher performance system that claims to transmit as many as 80,000 events per second, to improve scalability.

8. Figures

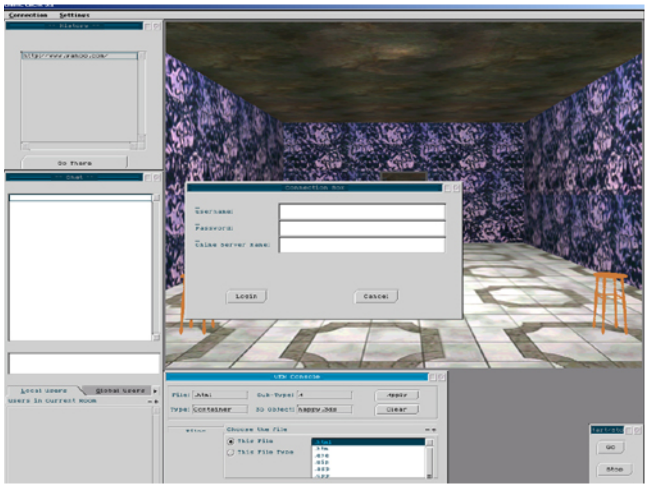
Figure 1 –Client requesting user credentials

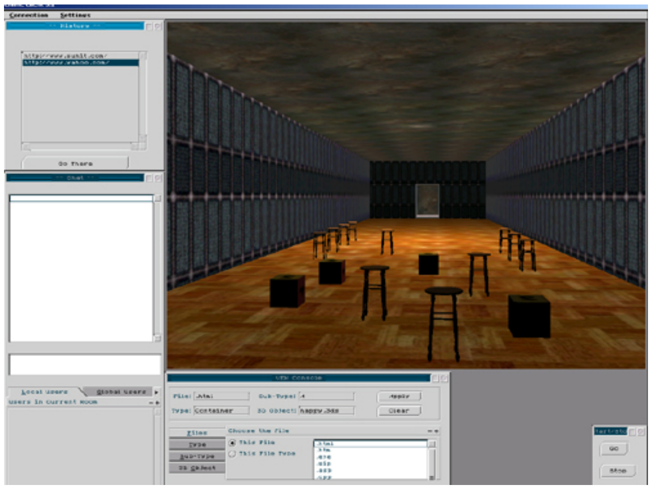
Figure 2 - Objects populating a typical room

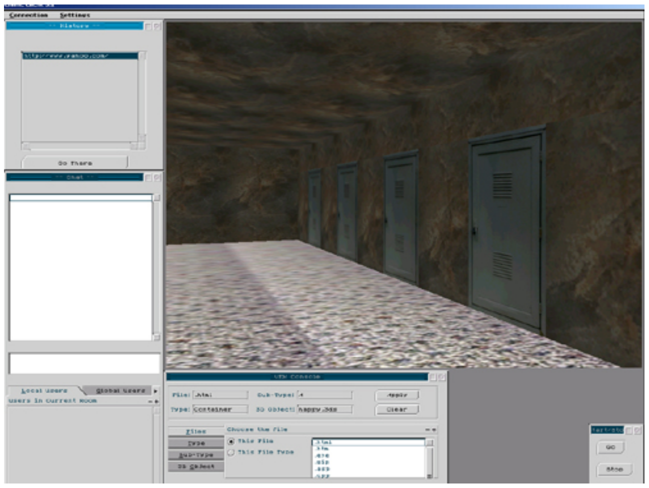
Figure 3 - Unlabelled doors ready to become links

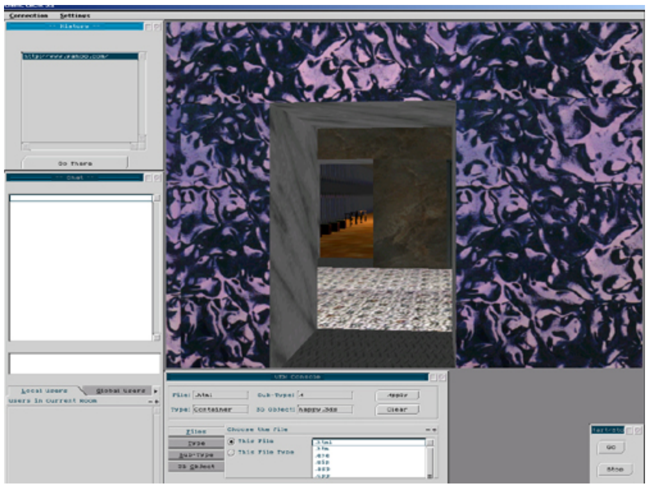
Figure 4 - Multiple rooms

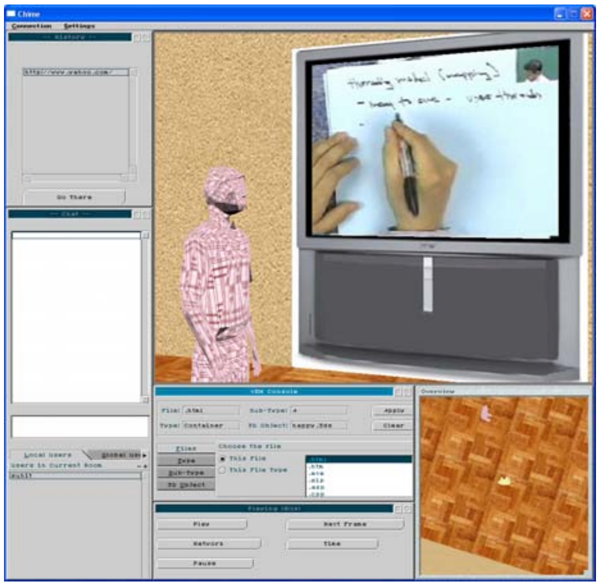
Figure 5 - Lecture video and team member

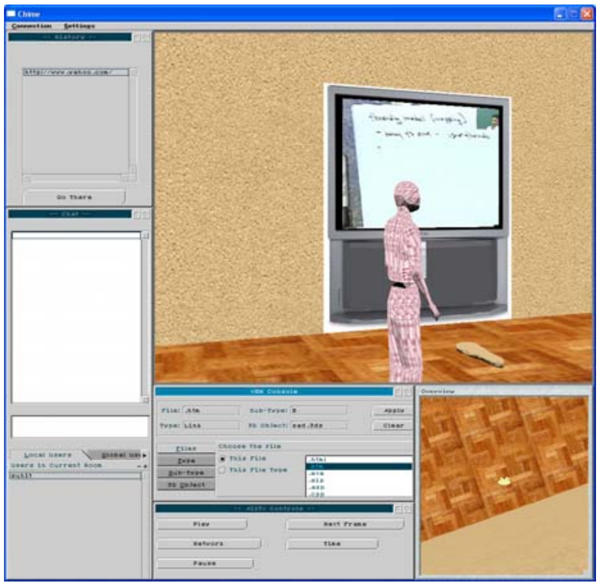
Figure 6 - Video orientation and perspective

9. Acknowledgements

We would like to thank John Kender and other members of his High-Level Vision Lab for their assistance in developing the video synchronization plugins. Peppo Valetto, Dan Phung and Matias Pelenur helped develop the second-generation AI^{2}TV plugin, presented separately in [21]. We would also like to thank other past and present members of the Programming Systems Lab, particularly Steve Dossick and Denis Abramov, for their work on earlier versions of CHIME. PSL is funded in part by National Science Foundation grants CCR-0203876, EIA-0202063 and EIA-0071954, and by Microsoft Research.

10. References

[1] R. Barzilay, N. Elhadad, K.R. McKeown, "Sentence Ordering in Multidocument Summarization", *Human Language Technology Conference*, Mar 2001.

[2] S.D. Benford et al., "VR-VIBE: A Virtual Environment for Co-operative Information Retrieval", *Computer Graphics Forum*, 1995.

[3] C. Bouras, A. Philopoulos, T. Tsiatsos, "e-Learning through Distributed Virtual Environments", *Journal of Network and Computer Applications*, Academic Press, July 2001.

[4] C. Bouras et al., "An Educational Community Using Collaborative Virtual Environments", *ICWL 2002*: 180-191.

[5] T. Capin et al., "Avatars in Networked Virtual Environments", John Wiley, 1999.

[6] P. Caplan, "You Call It Corn, We Call It Syntax-Independent Metadata for Document-Like Objects", *The Public-Access Computer Systems Review*, 6(4):19-23 (1995).

[7] A. Carzaniga, D.S. Rosenblum, A.L. Wolf, "Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service", *19th ACM Symposium on Principles of Distributed Computing*. July 2000.

[8] T. Daradoumis, F. Xhafa, J. Manuel Marquès, "Evaluating Collaborative Learning Practices in a Virtual Groupware Environment" *CATE*, 2003.

[9] P. Dillenbourg, "What do you mean by collaborative learning?" *Cognitive and Computational Approaches*, Oxford: Elsevier, 1999, pp. 1-19.

[10] J.C. Doppke, D. Heimbigner, A.L. Wolf, "Software Process Modeling and Execution within Virtual Environments", *ACM Transactions on Software Engineering and Methodology*, 7(1):1-40, Jan 1998.

[11] S.E. Dossick and G.E. Kaiser, "CHIME: A Metadata-Based Distributed Software Development Environment", *Joint 7th European Software Engineering Conference and 7th ACM SIGSOFT International Symposium on the Foundations of Software Engineerin*g, Sep 1999.

[12] S.E. Dossick, "Groupspace Services for Information Management and Collaboration", *PhD Thesis, Columbia University Department of Computer Science*, CUCS-001-2000, Nov 2000.

[13] S. Fiedler, M. Wallner, M. Weber, "A Communication Architecture for Massive Multiplayer Games". *NetGames*, 2002.

[14] G. Kaiser et al., "An Approach to Autonomizing Legacy Systems", *Workshop on Self-Healing, Adaptive and Self-MANaged Systems*, Jun 2002.

[15] G. Kaiser and G. Valetto, "Ravages of Time: Synchronized Multimedia for Internet-Wide Process-Centered Software Engineering Environments", *3rd ICSE Workshop on Software Engineering over the Interne*t, June 2000.

[16] J. Liu, B. Li, Y.Q. Zhang, "Adaptive Video Multicast over the Internet", *IEEE Multimedia*, 10(1):22-31, Jan/Feb 2003.

[17] T. Liu and J.R. Kender, "A Hidden Markov Model Approach to the Structure of Documentaries", *CVPR '00 Workshop on Content-Based Access of Image and Video Librarie,* 2000.

[18] T. Mansfield et al., "Evolving Orbit: a progress report on building locales", *Group97*, Nov 1997.

[19] M. Okada, H. Tarumi, T. Yoshimura, "Distributed virtual environment realizing collaborative environment education", *Symposium on Applied Computing archive, Proceedings of the 2001 ACM symposium on Applied computing*, Las Vegas, Nevada, United States Pages: 83 – 88, 2001.

[20] C. Oliveira, X. Shen, N. Georganas, "Collaborative Virtual Environment for Industrial Training and e-Commerce", *Workshop on Application of Virtual Reality Technologies for Future Telecommunication Systems, IEEE Globecom 2000 Conference*, Nov.-Dec. 2000.

[21] D. Phung et al. "Using Workflow to Optimize QoS for Collaborative Client Video Synchronization" *Tech Report, 2004*, Computer Science Dept., Columbia University – TR# cucs-009-04.

[22] E. Prasolova-Førland, "Supporting Social Awareness in Education in Collaborative Virtual Environments", *International Conference on Engineering Education*, 2002.

[23] E. Prasolova-Førland, "Supporting Awareness in Education: Overview and Mechanisms", *ICEE*, 2002.

[24] S. Singhal, M. Zyda, "Networked Virtual Environments: Design and Implementation", *ACM Press*, 1999.

[25] G. Valetto, G. Kaiser, G.S. Kc, "A Mobile Agent Approach to Process-based Dynamic Adaptation of Complex Software Systems", *8th European Workshop on Software Process Technolog*y, June 2001.