The Berkeley Continuous Media Toolkit

Brian C. Smith', Lawrence A. Rowe', Joseph A. Konstan3, Ketan D. Pate12

1Dept. of Computer Science 2Computer Science Division 3Dept. of Computer Science
Cornell University University of California
Ithaca, NY Berkeley, CA 94720
bsmith@cs.cornell.edu { larry, kpate } Q2cs.berkeley.edu

University of Minnesota
Minneapolis, MN 55455
konstan@cs.umn.edu

ABSTRACT

Conducting research on continuous media applications is hard, in part because it is necessary to have a complete multimedia system running to be able to evaluate changes to any component such as synchronization, network protocols, or encoding. The Berkeley Continuous Media Toolkit provides customizable support for an entire multimedia pipeline. This demonstration will illustrate several applications and research projects built using CMT while showing how CMT is used to build applications and experiment with innovative multimedia systems research.

KEYWORDS

Multimedia toolkits and frameworks, continuous media, distributed multimedia.

INTRODUCTION

Over the past four years, the Plateau Research Group at the University of California, Berkeley, has been developing tools for building distributed continuous media applications. The fruit of this effort is the Continuous Media Toolkit (CMT [1]), a comprehensive toolkit and framework for creating experimental CM applications. As a toolkit, it contains a set of commonly used tools needed to create CM applications. As a framework, it defines a set of architectural guidelines for constructing these applications. CMT can be used as a platform for building applications and for conducting multimedia research. CMT has been used for research on synchronization (including the development of a synchronization toolkit), network transmission, video-on-demand servers, new media types, and real-time software decoding [2, 3, 7]. CMT has also produced several important “spin-off” technologies, including the Berkeley MPEG player [6], TclProp, a data propagation constraint programming environment for Tcl [4], and Tcl-DP, the distributed programming extension to Tcl/Tk [9]. This demonstration will show the community how to use CMT as a framework for research on elements of continuous media systems (e.g., new network transmission protocols). The rest of this document explains the CMT framework and provides an overview of the demonstration.

THE CMT FRAMEWORK

Doing research on continuous media applications is hard. Researchers are faced with multiple difficult problems, including the need for high performance hardware or software encoding and decoding, inter-stream synchronization, resource management, distributed programming, novel network protocols, application adaptivity, and user interface management. From a researcher perspective, evaluating a solution to any one of these problems means having some solution to all of them, since the usefulness and quality of such a solution is ultimately determined by how it looks to the end user and how well it interacts with the rest of the system. Thus, researchers are faced with an ominous task when experimentally evaluating solutions.

To simplify this research problem, we developed a toolkit and framework called CMT. CMT, an extension to Tcl/Tk [5], provides a suite of applications on which new solutions to research problems can be tested and a toolkit for building new applications. For example, suppose you have developed a new network protocol for video delivery that you want to test. The CMT architecture allows you to easily insert your protocol into the CMT distributed video playback application CMPlayer [8] by changing just a few lines of code. You can then test your network protocol in a real, distributed system, evaluating the interactions with other system components and the quality of the transmitted video on the screen. Similarly, you can evaluate your network protocol with any CMT-based distributed video applications.

CMT arms the researcher with three types of tools. First, it provides a set of commonly used software tools, such as codecs, network delivery mechanisms, and user interface 451
widgets. These tools include hardware/software codecs and players for MPEG and Motion JPEG video, software interfaces for uncompressed audio, built-in support for several best-effort network protocols, and a variety of tools for scheduling and memory management. Second, it provides a generic data model for representing CM data. This data model accommodates both live- and recorded-media applications, and it is easily extended to include new data types. Third, it provides a programming model for constructing CM applications. The programming model uses a data-flow model where video data is passed between inter-connected objects. Formally, a programmer simply draws the graph for the data-flow and each node and edge in the graph translate into roughly one line of code in a Tcl script.

These mechanisms make it easy to build sophisticated applications using CMT. For example, a simple video file server, with logging and security features, is about 75 lines of code. Distributed video playback can be implemented in as few as 10 lines of code. But more important for researchers, any given element in an application can be easily replaced. Returning to the network transport example, the transport protocol is represented as a node in the data-flow graph. To use a new transport protocol, one would simply replace the appropriate object(s) in the graph of an existing application. This translates to substituting one or two lines of easily locatable code.

THE DEMONSTRATION
The demonstration has two major objectives: showing some of the applications and tools that we have built using CMT and showing how new objects and applications can be constructed.

The applications and tools to be demonstrated include:

- The CM Player – a distributed audio/video player.
- Applications of the command stream extension that allows Tcl commands to be synchronized with other media.
- A demonstration of the use of data value propagation for developing multimedia user interfaces.
- An intra-network compressed domain real-time MPEG to JPEG transcoder.
- The SynchMe synchronization research toolkit including a demonstration of skew control implemented as a set of temporal constraints using CMT.

REFERENCES


