AUTOMATED DATACAST APPLICATION DEVELOPMENT

P. Caputo, D. Catapano, N. Nandhakumar, G. Thomas
Triveni Digital, Inc., USA

ABSTRACT

This paper describes a method for greatly enhancing the value of data broadcasting (datacasting) by automating development of datacast applications. This method provides tools for a subject matter specialist to organize the content and specify the desired receiving sites for various collections of content. The content is then automatically broadcast to the specified sites and stored on servers there. Moreover, a web-based search/navigation interface is automatically generated for end users at these sites to access the broadcast content (using ordinary web browsers). The subject matter specialist who develops the application does not need any expertise in data broadcasting or web site design.

INTRODUCTION

Data broadcasting via digital TV offers a very efficient way to deliver rich multi-media content to many recipients simultaneously, often just using excess bandwidth in a broadcast stream. One good example of an application is the distribution of educational materials such as videos, slide shows, electronic books and magazines, together with associated teaching guides, sample quizzes, background reading, etc., to schools.

Delivering such content in the form of physical tapes and paper requires high reproduction, packing, shipping, unpacking, etc., costs. These costs can be avoided by making the content available on centralized web sites for on-line access, but this is often not a good approach, since it requires very high bandwidth Internet connections to the schools. It is typically much more economical to deliver the content via datacasting to servers in the schools, where it can be accessed on demand over a local area network (LAN).

The method described in this paper greatly reduces the effort required to develop an end-to-end data broadcast application of this sort. Moreover, it works equally well whether the data is delivered via satellite, cable, terrestrial broadcast, or other broadcast medium, or when it is delivered across a very wide region through a hybrid network such as distribution by land line or satellite broadcast to TV stations and broadcast by terrestrial TV to end user sites.

OVERVIEW OF DATACASTING AND APPLICATIONS

Digital television (DTV) technology not only supports high definition digital TV (HDTV) and the broadcasting of multiple standard definition TV (SDTV) channels in a single broadcast stream, it also supports datacasting – the insertion of data into a broadcast stream along with audio and video.
How Datacasting Works

The various DTV standards (e.g., DVB, ATSC, ISDB-T) are all based on the MPEG-2 transport stream concept defined in the MPEG-2 systems standard (1). Audio streams and video streams are carried in small transport stream packets that are multiplexed (interleaved) into the overall broadcast bit-stream. Each packet contains a packet ID (PID) field that identifies which audio or video stream it belongs to, so that DTV receivers can pick out the packets corresponding to the audio and video streams for the TV channel a viewer has selected.

For datacasting, data items are broken up into transport stream packets and interleaved into the broadcast bit-stream along with the audio and video, using different PIDs to distinguish data streams from each other and from the audio and video for the TV programming. Any type of digital data can be carried this way, including files, IP streaming media, and other types of IP traffic. A data server at the head end encapsulates the data into transport stream packets and sends them to multiplexers which merge the data packets with the audio and video packets in the broadcast stream. Data-capable receivers extract the packets and reconstruct the data items. Data broadcast standards such as the DVB data broadcast specification (2) and the ATSC data broadcast standard (3) define the data encapsulation format, thereby enabling interoperability among data servers and receivers from different implementers.

Types of Datacast Applications

There are a number of different ways to classify datacast applications:

- Enterprise-to-consumer vs. enterprise-to-enterprise
- Streaming content vs. discrete data files
- Coupled to TV program vs. unrelated to TV program

For example, interactive TV uses datacasting to enhance the TV viewing experience by providing extra information related to the show, such as game/player/team/league statistics during sports events. This is typically a consumer application sending data files coupled to the TV program.

As another example, some school districts are starting to use datacasting to deliver both live lectures and other educational materials to the schools in their district. This is an enterprise application using both streaming content and data files that are completely independent of ordinary TV programming.

This paper focuses on enterprise-to-enterprise applications that deliver files which are independent of the underlying TV programming.

Educational and Training Applications

A very promising application for datacasting is support for educational and training programs at all levels – primary school, secondary school, college, adult, etc. As noted above, both live videos and all types of data files can be delivered this way, including pre-recorded video clips, slide show presentations, supplementary reading materials, teaching guides, sample quizzes, etc. Often such materials must be distributed over a geographical area roughly corresponding to the broadcast footprint of a single TV station, making this an ideal application for terrestrial DTV datacasting.
Datacasting vs. Internet Access

To ensure appropriate use of datacasting, it is important to understand when datacasting is a better solution than on-line internet access. Relevant factors are:

- Is it necessary to deliver rich, multi-media content?
- Is the content destined for many receiving sites?
- Does the content need to be updated fairly frequently?
- Are some receiving sites in out-of-the-way places with no broadband access?

For example, even if schools have a T1 Internet link (1.44 Mbps capacity), it is usually not a good idea for classroom teachers to play streaming video for their classes directly from a remote web server. To get acceptable video quality, the streams must be encoded at 300-500 Kbps or more, and as few as 3 or 4 streams will overload the T1 line. Moreover, if teachers at many schools play such videos simultaneously, it puts a tremendous load on the web server providing the video streams. It works somewhat better for individual teachers to download video clips ahead of time and stream them from machines in their own classrooms, but this still puts a heavy load on the Internet connections and remote servers. It makes more sense to use datacasting to pre-position the more popular videos on file servers in the schools and stream them to the classrooms on demand over the 100 Mbps LANs in the schools. This eliminates long download times and jerky playback.

AUTOMATED DATACAST APPLICATION DEVELOPMENT

Developing a datacast application to deliver file content takes more than just dumping the content into the broadcast stream at the head end and extracting it at receivers. The content needs to be packaged up and delivered to the receivers in a form that enables end users to locate the content they want easily, and lets them see content items in the context of other related content items. This can be done manually by developing a web site around the content, and datacasting the files that make up the web site. Each data receiver can then extract from the datacast the content of interest and end up with a local web site for users at that location. However, this approach has problems:

- Typically different receiving sites will need different subsets of the content, and these subsets will overlap in complex ways. It is difficult to design a coherent web site which does not result in broken links at the local web sites when each receiver extracts only part of the content.
- It is even harder to update such local web sites selectively without introducing broken links.
- The design and management of such a web site is very time intensive, and the staff must be highly skilled, with expertise in web design and datacasting as well as the subject matter.

A toolkit is needed that produces more robust datacast applications more quickly and easily.

Overview of an Automated Approach

The approach proposed in this paper allows content to be packaged up and targeted to receivers by subject matter specialists who do not need any expertise in web design or datacasting. They only need to understand the attributes of the content items and the logical interrelationships among them. They can attach descriptive metadata to content items to form “assets.” They can group assets into
“packages” and associate descriptive metadata with the packages. They can group packages into higher level packages, forming any number of package trees which may overlap in arbitrary ways. They can designate which assets and packages are intended for which groups of receivers.

The underlying infrastructure then schedules the broadcast of the assets and packages automatically, with targeting information so each receiver knows what assets and packages to extract.

At each receiving site the package hierarchies automatically show up as navigational paths in a local web site, and the descriptive metadata are not only available for display, but are also available as search criteria for locating content in the local web site.

The Automated Approach as Seen by the Content Specialist

The tools for the content specialist at the head end support the following operations:

- Creating, modifying and deleting definitions of individual receivers and groups of receivers
- Creating, modifying and deleting assets and packages
- Creating and deleting assignments of assets and packages to receivers and groups of receivers

Note that “creating” or “deleting” an asset does not mean creating or deleting the underlying content item. It means adding or removing identification of the content item and its metadata to the system.

Assigning assets and packages to receivers causes them to be broadcast to those receivers. If the underlying datacasting system supports adequate in-band signalling to the receivers, such as that described in a recent paper by Catapano and Thomas (4), then removing assignments of assets or packages to receivers causes the receivers to delete them from their local server.

Figure 1 below shows an example of a user interface that can be used to add packages to an existing higher level package. The user has put some packages on a temporary “clipboard” (at the bottom), and has just finished copying the Biography package from the clipboard to the existing History package (at the top), joining the four other packages that were already in the History package.

**Figure 1. Adding a Package to a Higher Level Package**

Presented at BroadcastAsia 2004 conference, June 16, 2004, Singapore
The Automated Approach as Seen by End Users

Figures 2 and 3 below show how information might be displayed to an end user at a receiving site.

Figure 2. Viewing Collection of Packages on a Receiver

Figure 3. Viewing Description of an Individual Asset

Presented at BroadcastAsia 2004 conference, June 16, 2004, Singapore
Figure 2 shows the collection of all top level packages at the site. A very similar page can be used to show the contents of any package or the results of a search. The user can click on any icon to see the contents of the corresponding package, and continue drilling down in this way to reach the individual assets. When the user clicks on an icon representing an individual asset, a display similar to that in Figure 3 would be shown. From this page the user can click on the button near the bottom to launch the asset (which in this example would mean play the QuickTime movie).

It is important to note that these displays are generated automatically from the asset and package structure defined by the subject matter specialist who developed the application, using HTML templates built into the toolkit. The specialist does not have to design the web pages.

IMPLEMENTATION ASPECTS

The recommended user interface approach for such an automated datacast application development toolkit is a web-based interface for both the subject matter specialists who develop applications and the end users at receiving sites. This allows them to use their favorite web browser without being tied to any specific computing platform, and also facilitates remote access to the functionality.

The recommended approach to providing the HTML pages for the end users to access the datacast content is to generate the pages dynamically at the receiving sites, using the descriptive metadata that is datacast as part of the assets and packages, rather than pre-generate the pages and datacast them to the receiving sites. This approach requires extra software to be installed on the receivers, but it requires less bandwidth in the broadcast stream, and it is far more robust in dealing gracefully with situations where data is lost during transmission.

System Architecture

For ease of implementation, the toolkit can be based on an underlying datacasting system, which would typically consist of a scheduling tool, a data server, and data receivers. The toolkit itself can consist of two components, a Package Station acting as an enhanced scheduling tool, and a Presentation Engine acting as an enhanced data receiver, as shown in Figure 4 below. The Package Station allows the subject matter specialist to organize the content and assign it to receivers, and it then schedules the broadcast of the content and associated metadata. The Presentation Engine generates the web page displays at the receiving end.

![Figure 4. Architecture for Automated Datacast Application Development Toolkit](image-url)

The system can also include a Customization component, not shown in Figure 4, that allows definition of customized metadata fields and customized HTML templates for the web pages at the receivers, in order to provide specialized Package Stations for different application domains.

Presented at BroadcastAsia 2004 conference, June 16, 2004, Singapore
Implementation Approaches

The Package Station for the subject matter specialist can be based on an ordinary database system for holding the information about the assets, packages, and receivers. This database contains the descriptive metadata for each asset and package, and the metadata defining the relationships among the assets and packages. It also contains the definitions of receivers and receiver groups, and the assignments of assets and packages to receivers. The operations described above for maintaining all this information are provided by a web application.

The Presentation Engine for the end users at receiver sites can also be based on an ordinary database system that holds the received metadata for the assets and packages. This includes pointers to the locations on the local disk of the received content items represented by the assets. A web application generates the navigation and search pages dynamically from the database metadata, using pre-defined HTML templates that are sent in the broadcast. When new or updated assets and packages arrive, the content items are stored on disk, and the metadata in the database is updated.

FIELD EXPERIENCES

An application development toolkit based on these principles has been developed as an adjunct to Triveni Digital's SkyScraper™ data broadcast system. It has been deployed in the field for some time, with a very positive response from users. With this toolkit it is easy to develop a small demonstration datacast application in an hour or so, assuming the underlying content items already exist. Much larger applications can be developed in just a few days, rather than many weeks.

CONCLUSIONS

As datacasting becomes an increasingly viable way to distribute information to large numbers of people at large numbers of sites simultaneously, the ability to develop datacasting applications quickly and easily becomes increasingly important. The approach described in this paper provides a powerful and flexible way to do that.

REFERENCES


ACKNOWLEDGEMENTS

The authors gratefully acknowledge the contributions of our colleagues to the development of this technology, including especially Jonathan Schembor, Brad Holcombe, Richard Gong, Sean Devine, Rangarajan Padmavijayam, and Tom Weimer.

Presented at BroadcastAsia 2004 conference, June 16, 2004, Singapore