

THE LIBERATE TECHNOLOGIES TV NAVIGATOR FOR DTV: A THIN WEB-CENTRIC CLIENT FOR DIGITAL TELEVISION

Dr Stewart Bryant and Raymond Drewry

Oracle Corporation and Liberate Technologies

ABSTRACT

The Liberate Technologies TV Navigator for DTV is a thin, distributed, web-centric client and application suite for digital television set-top boxes (STB). This paper describes the scalable client and server processing architecture of DTV Navigator, demonstrating how the low memory, yet highly flexible system design was achieved. This paper shows how the navigation and application suite are layered onto the client HTML and JavaScript engines, and the necessary JavaScript extensions to operate in the television environment. An applications suite provides TV optimised access to the common web applications such as mail, chat and information tickers. The design approach used by DTV Navigator provides the network operator with a secure and yet fully customisable user interface that can be quickly adapted by web programmers to meet new product and marketing opportunities. This approach also allows DTV Navigator to support the cross-linking between the web and broadcast television, so that the network operator can make optimal use of these two distinct media types.

INTRODUCTION

The rapid convergence of digital television with other information services, such as the Internet, requires a client runtime and authoring environment that can operate effectively and efficiently within constraints imposed by television set-top box. This paper describes the approach to this problem used by Liberate Technologies' TV Navigator for Digital Television.

SET-TOP CONSTRAINTS

Television set-top box (STB) software runs in a highly constrained environment. The STB hardware design is constrained by cost and reliability. As digital television media develops, the STB software needs to aspire to the capability of that are normally found on a PC or network computer, and yet needs to do so with no intervention on the part of the user.

The fundamental hardware driver is low cost, which translates into slow to medium processing power and a low memory budget. The current absence of a hard disk means that all processing must take place in physical rather than virtual memory. This memory constraint is outside the experience of many applications writers.

Unlike a PC, the expectation is that there will be no system maintenance by the customer, there will be no customisation, and configuration will be minimal. Furthermore, the need to recover the costs of subsidised deployments means a far greater service life than is normal in the PC environment.

Because the STB provides continual access to high value media, and because of the commercial transactions that will be conducted from the STB, security is a particular issue.

NETWORK CONSTRAINTS

Although digital television networks deploy high bandwidth communications paths for audio-video content, the interactive channel characteristics vary widely from deployment to deployment. The network characteristics, range from systems with high performance bi-directional interactive channels (for example ASDL over telephone networks or MCNS over CATV) through to systems that provide low bandwidth dial-up modems. The synchronising nature of

television programmes tends to mode-lock interactions causing extreme peak to mean ratios of activity. Furthermore, both commercial and strategic imperatives require the system to always be operational for television viewing regardless of any networking problems associated with the interactive channel. This requires the STB software to be capable of operation in the absence of the interactive network and the applications to be written to emphasise interaction with the media rather than with their environment.

AUTHORING ENVIRONMENT

The digital television market is at a critical phase of its development, with a number of competing authoring environments vying for supremacy. A critical success factor will be the speed with which new and pre-existing applications and services can be made available to viewers. The WWW authoring tools for HTML and JavaScript, facilitate rapid content deployment by:

- taking advantage of existing services and content
- taking advantage of existing tools
- taking advantage of existing trained authors
- providing known error-tolerance
- allowing re-use of existing web based back-end infrastructure

It was for these reasons that Liberate chose to use HTML and JavaScript as the authoring environment for DTV Navigator.

NECESSARY AND SUFFICIENT APPROACH

The core of the DTV Navigator STB client is an HTML and JavaScript engine which provides the display and application execution environment. The JavaScript engine has been enhanced to support additional object types necessary to operate an integrated television – Internet environment.

The key to the DTV Navigator design was to take a “necessary and sufficient” approach to the design. This was achieved by:

1. Distributing the client between the STB and a number of specialist support servers located in the network.
2. Minimising the number of MIME types supported on the STB to a small but sufficient set. A transcoding server converts other popular MIME types to this canonical set.
3. Dispensing with the native applications normally built into browsers (i.e. the browser itself, and mail) and replacing them with applications written in HTML and JavaScript.
4. Creating a storage hierarchy in which local volatile storage, local non-volatile storage and broadcast data are viewed as cached copies of requested data.
5. Minimising the system state and storage requirements.

As more powerful STBs become financially viable, these constraints can gradually be relaxed to allow additional STB native MIME types and more powerful client application languages such as Java. As this evolution happens, the DTV support servers will play an important role in helping legacy STBs to access new applications.

CLIENT-SERVER ARCHITECTURE

The architecture of the Liberate TV Navigator for DTV is shown in Figure 1. The distributed client has a set of client side and server side components. On the client side is the base STB hardware with its drivers and operating system. Layered on top of this is the HTML and JavaScript engine. This engine runs a “resident” application written in HTML and JavaScript, which in turn runs the user applications.

The server components of the distributed client consist of the DTV services server, supported by a number of specialist servers.

The DTV services server hosts the transcoding server, the proxy server and the security server. A client requests a URL from the proxy server, using HTTP. This proxy server holds a cache of translated material, and if possible responds to the STB with a pre-translated response. If the material is not in the transcoding cache, it is requested and the response directed to the transcoding server. The transcoding server parses the HTML to process any errors, translates MIME types to the canonical form, scales images and lays out the page. This layout information is included in the HTML sent back to the client. The client thus makes conventional HTTP requests and receives conventional HTTP responses pre-processed to minimise the amount of client side processing required. Both client and server cache these responses for later re-use.

If the STB makes an HTTPS request, this is staged through the security server. The security server acts as a relay with separate client side and server-side session keys. This is necessary to allow the transcoding server access to the content in the clear so that it can perform the pre-processing operations outlined above. The proxy server does not cache secure translated material. Some information is so secure that this break in the end to end encryption path is unacceptable. Provision is therefore made for ASCII and binary data to be routed directly to the STB. Information fetched by this path bypasses the HTML and JavaScript engines, and must be processed directly by the JavaScript application code

The three components of the DTV support server can be run on separate hardware platforms. It is recommended however, that the transcoding server and the security server run on the same hardware to avoid secure information appearing on the network in the clear. The DTV services server is stateless (apart from the HTTPS session keys). The use of a virtual IP router between the STB and the DTV services servers allows the STB to be configured with a single DTV services server IP address whilst load-balancing and fail-over happen transparently. Note that to save memory the STB does not have a DNS client. When requesting content through the DTV services server, this server performs the necessary DNS services on behalf of the STB. However to address the DTV services virtual server and any directly connected HTTPS servers the STB must use the 32bit IP address, rather than a name.

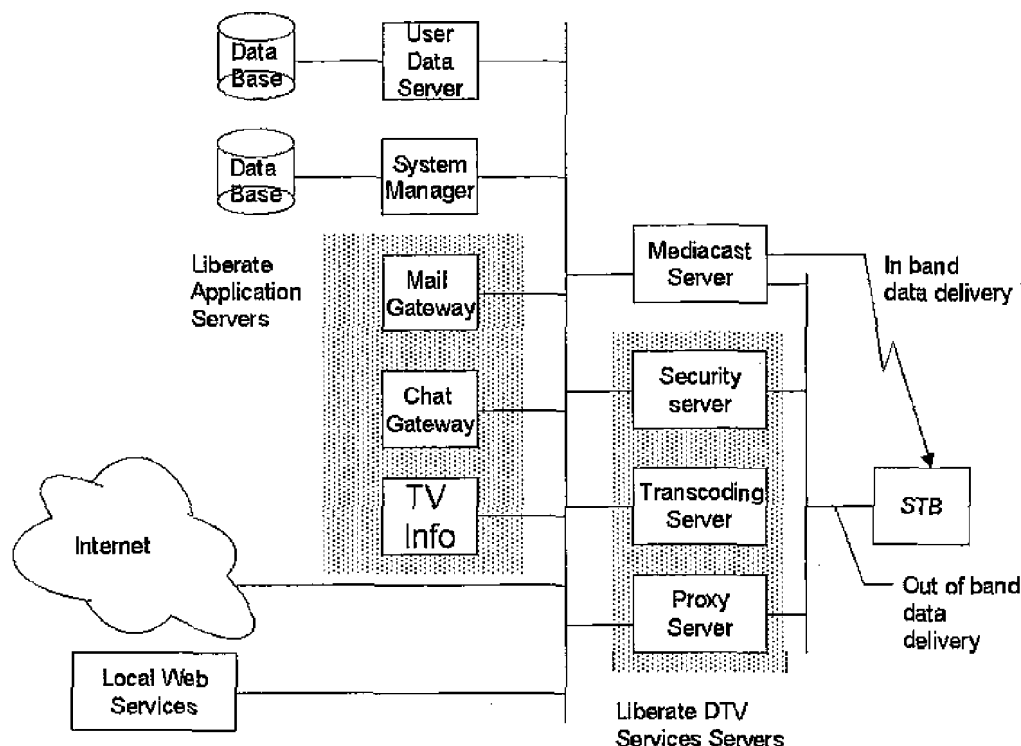


Figure 1: Liberate DTV Navigator End-to-end Architecture

When content requests will be synchronised by a television-programming event, and will therefore be concurrently requested by many users, it can be advantageous to transmit this information using MPEG data broadcasting, or multicast IP. The Mediacast server is programmed to build data carousels using a scripting language based on the CDF language. It transmits these carousels to the network via either a broadcast network using a DVB ASI interface or via IP Multicast over a variety of network types (Ethernet, MCNS etc). The Mediacast server masquerades as a STB to the DTV services server, using its facilities to pre-process the content. The route by which information arrives at the STB is transparent to the JavaScript application, thus carouselled material received from the Mediacast server appears to the application as if it were retrieved from the STB onboard cache.

The User Data Server supplements the STB's limited non-volatile storage. This server stores user configuration information and cookies. The User Data Server is a front end to a conventional relational database (normally Oracle 8). The User Data Server is therefore stateless, relying on the underlying database for reliability, availability and backup services. A CORBA interface to the User Data Server allows the service provider to manipulate the user data directly. This facility is useful in automatic provisioning of the user. It is also a useful tool for managing the user context by the customer service representatives in the call centre.

The TV Info server interfaces to the schedule provider and provides schedule information to the network. The interface to the schedule provider is described in XML making it adaptable to new input formats. The output to the network can be standard DVB SI, Liberate Technologies enhanced format, or a combination of both. The output may be broadcast, multicast, or explicitly fetched by the STB. Both the TV Info server and the STB may operate with conventional DVB SI. This makes the TV Info server optional, however the TV info server adds value in a number of ways:

1. By incorporating a schema along with the transmitted data it is able to support the dynamic extension of the schedule by making the data self-describing.
2. By compressing the schedule data, less bandwidth is consumed in broadcasting the schedule. The service provider may take advantage of this to reduce the required bandwidth or to reduce the carousel spin time.
3. Because the TV Info server supports a point-to-point mode, allowing the STB to directly request schedule information, network bandwidth may be saved by not broadcasting schedule information of minority interest.
4. Finally the availability of the schedule information on a server that may be addressed directly by the STB means that complex schedule requests may be supported server side where there is greater memory and a more powerful programming environment.

The Liberate Technologies servers are configured by a management server and are monitored using SNMP.

THE CLIENT ARCHITECTURE

The DTV Navigator for client architecture is shown in Figure 2. The core of the system is the run-time engine, which provides the basic HTML and JavaScript interpreters, the core additional JavaScript objects needed for this environment, and their operational infrastructure. This is system independent code that runs over a well defined Porting API. The STB designer must abstract the external system environment and the STB hardware and run time implementations to provide this Porting API to the DTV Navigator client software. Typically, the porting interface will provide interfaces to the real-time operating system, the network, the video sub-system, the MPEG sub system, the audio sub-system and the user input. Additional JavaScript objects may be added to the basic run-time engine with the extender system API.

The TV Master Application transforms the basic HTML and JavaScript engines into the applications environment that the external applications will use. The TV Master Application can be thought of as transforming the raw engines into the TV equivalent of the desktop. Liberate produce a number of common applications such as TV Mail, TV Chat and TV Ticker.

LIBERATE TV EXTENSIONS TO JAVASCRIPT

Navigating in the television environment requires extensions to JavaScript so that the application can efficiently interact with new objects. Three conflicting constraints effect the design of these objects:

1. Any new object adds additional code to the JavaScript interpreter, this conflicts with its minimum memory design constraint.
2. An object that is provided by the JavaScript interpreter executes the functionality faster than the equivalent functionality written in JavaScript and executed by the interpreter.
3. Because digital television is a new technology, there are no two systems with an identical architecture. The JavaScript extensions need to cut across these differences to provide the required functionality independently from the underlying infrastructure.

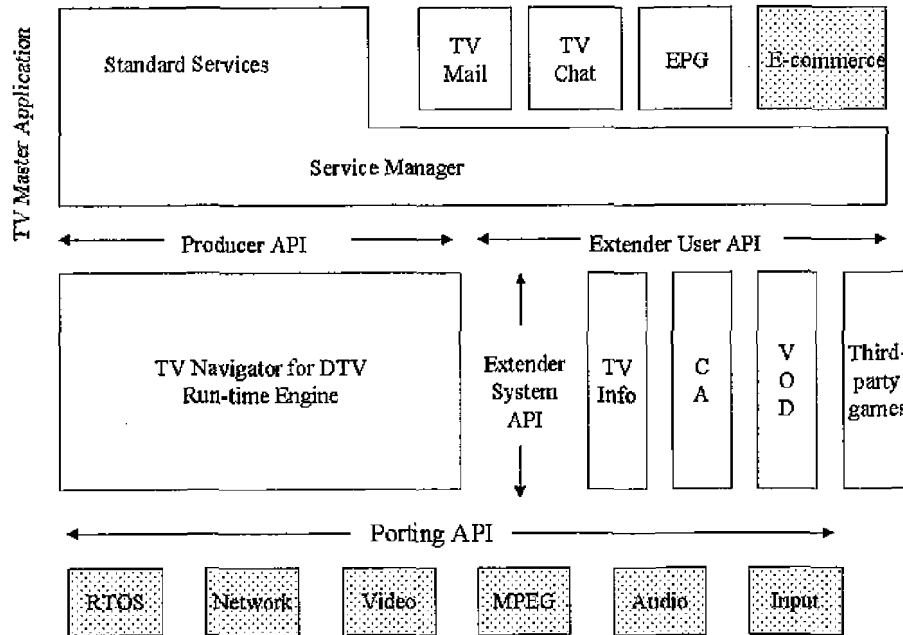


Figure 2: TV Navigator for DTV Client Architecture

Liberate technologies DTV Navigator adds approximately 30 objects to those found in standard JavaScript. These objects fall into a number of categories:

1. **TV viewing objects:** to allow the user to find out what programmes are/will be broadcast, to purchase those programmes, and to view them.
2. **Storage objects:** to allow applications to store and retrieve state information with various levels of latency, persistence and security. The operation of these objects allows the remote management of the STB users environment.
3. **Network Objects:** to allow the application to request and retrieve information from the network using point-to-point, closed user group and open access technology.
4. **Web objects:** to allow access to user history, and allow the application to write text to table cells. The dynamic table support allows the de-coupling of the layout process (performed on the transcoding server) from the composition of text output by the JavaScript application.
5. **Management objects:** to support multiple user environments for different family members (parents, older child, younger child...) or different personal moods (news hound, couch potato...).
6. **System specific objects:** Each system instance has a number of non-generic controls that must be accessed by JavaScript (e.g. LED displays). These are mapped directly to the application through the indicator object, and are transparent to the JavaScript engine.

Standard JavaScript provides domain security, however this is not sufficient to prevent this commercially valuable interface being subordinated or spoofed by third parties. DTV Navigator has therefore enhanced the security model by applying access control list protection to critical JavaScript objects (i.e. TV tuning, user management, shutdown etc).

THIRD PARTY EXTENSIONS TO JAVASCRIPT

In addition to the core TV extensions built into JavaScript by Liberate, the Extender System API is provided to allow the partitioning of enhanced functions that are needed for custom builds, or that vary significantly from one system implementation to another. Examples of these are TV Information and Conditional Access. The TV information system is incorporated in this way to allow for the changes between DVB and ATSC, and to allow customers to develop their own information profile, possibly enhanced with information carried in the Liberate compressed format. The conditional access market is highly fragmented, and often has proprietary enhancements, and so again it is not included in the system independent core. This API is also provided to partner organisations to allow them to produce JavaScript objects that can be linked into a custom build. This enables other vendors to integrate their functionality into the STB environment and build enhanced JavaScript applications. The first examples of this to be deployed are a gaming JavaScript object from the UK company TwoWayTV and video-on-demand JavaScript objects from a leading VOD vendor.

THE TV MASTER APPLICATION

The first JavaScript application loaded by the STB is known as the TV Master application (TMA). This is the application that converts the bare JavaScript and HTML engine into an environment that will host the STB applications. The TMA provides a secure set of objects and services that can be used by the user applications. In addition to initialising the application software environment, the TMA connects to, and registers, with key handlers. It also connects to data streams, handles global key events (such as keys that must operate all the time, e.g. channel up/down), and displays overlay screens (e.g. channel number).

The TMA is also responsible for validating and logging on individual users, the presentation and registration of new services, and parental control of access to both TV and web based services. This application stays resident in memory throughout the current operational session and holds the context of the session.

The TMA is written in a mixture of HTML and JavaScript and is a library of example services that the service provider can customise. This allows the service provider rather than the software vendor to "own" the television screen, and hence the relationship with the consumer. It also allows the service provider to provide one or more user interfaces that properly reflect the culture and sophistication of the particular viewer. The standard services provided are:

1. Status: displays status of operations underway, for example web page loading.
2. Users: manages multiple user accounts
3. Channel: which controls the TV channel selection
4. Volume: which controls the STB audio
5. ATVEF: which handles the ATVEF triggers and reverse path requests
6. Favorites: which is a bookmark service for web pages
7. Home: which takes the user to a control panel from which the user can access the navigator services whilst watching television.
8. Menu: which invokes a set of common applications such as web-browser, TV guide, User, Mail and Chat.

The capability of viewing television programs is the most critical application provided by the STB. The reliability of traditional television systems has created a user expectation of instant access to a system with a 100.0% availability. If the resident application and the television service navigator were loaded from the network each time the STB is re-booted, there is a chance that the system would not live up to this reliability expectation. This difficulty is addressed by providing the resident application and TV navigator in non-volatile storage on the STB rather than loading it over a network. The storage requirements of these applications are not excessive (typically 50 KB); and are normally dominated by the graphical bitmaps. DTV Navigator can provide significant functionality without a two-way network in place. A particularly useful approach in limited memory configurations is therefore to partition the resident and navigator application into two components. A basic user interface loads from non-volatile storage and is available for

immediate use by the viewer. This is enhanced by other components, which silently load across the network when that resource becomes available.

ACCESS AND SERVICE MANAGEMENT

Commercial and social constraints require the designer of a television browser to give significant attention to access management. Parental and commercial control must be provided for both web and TV activity. The DTV Navigator adds additional information to the distributed client-server protocol to screen requests against a set of constraints. These constraints are stored in the user data server and are loaded when the user logs on.

Additionally the navigator must provide support for service subscription management. This allows the system to advertise new services (such as news tickers, email access etc) and the user to register for them on an individual basis. Subscriptions can be either user or operator-driven, and by supporting them through the infrastructure and the resident application they may be displayed, billed and administered in a uniform way.

LAYERED APPLICATIONS

Liberate has written three client side applications as examples (TV Navigation, TV Chat and Web Browsing) and two client-server applications (Mail and Chat). In all cases, the client side is fully customisable by the service provider to tune the look and feel of the application.

A server acting as a gateway between the client and an IMAP mail server supports the mail application. In common with all the other servers, this gateway is stateless. The used approach is not to be confused with the HTML mail interfaces commonly found on the web. A pure HTML approach does not allow satisfactory integration between the mail application and other client applications. Furthermore had that approach been taken, all mail pages would have needed full processing by the transcoding server. This processing load is reduced by using a template method. The template needs processing by the transcoding server, (but in practise will be cached), but much of the data passes straight through to be displayed by the client using dynamic tables. The mail content itself still needs to be fully processed by the transcoder to allow the mail system to support multi-part MIME emails.

In a similar way to mail, the chat server provides a stateless gateway between client and an Internet Relay Chat server, with the application again using a template approach to minimise the load on the transcoding server. Both the mail and the chat applications are examples of true layered applications that demonstrate the ability of the infrastructure to support services written by third parties.

Ticker services (e.g. weather, stock reports, headlines) are implemented with a set of web page templates for display of the data, and a set of Mediacast data carousels that contain the information. The data received by the STB is collected by a JavaScript function, which uses it to populate the template pages (which may be in carousels themselves.)

CONCLUSION

This paper has described the Liberate Technologies TV Navigator for Digital Television. It has shown how this distributed client supports the use of HTML and JavaScript as an authoring environment to integrate the television and Internet environments. The necessary JavaScript and protocol extensions and some example applications were described.

