BBC

R&D White Paper

WHP 019

January 2002

Delivering Interactive DTV services to multiple target platforms

R. Bradbury, R. Cartwright *and* **T. Steele**

Research & Development
BRITISH BROADCASTING CORPORATION

BBC Research & Development
White Paper 019

Delivering Interactive DTV services to multiple target platforms

R. Bradbury, R. Cartwright and T. Steele

Abstract

The mature MPEG specifications for digital video and audio ensure the availability of interoperable equipment that simplifies the task of creating broadcast content in a cost-effective way. In contrast, no single technology has emerged as the common standard for multimedia services. Whilst many content providers have achieved the launch of bespoke multimedia services for the deployed target platforms, authoring multiple times is not a cost-effective long-term solution.

Initiatives such as DVB's MHP and ATSC's DASE are attempting to solve this problem by establishing multimedia standards for DTV. Regardless of their outcome, a legacy of already-deployed technologies will exist for delivery platforms that are either unable or unwilling to migrate. This paper describes the BBC's views on authoring and delivering interactive services to deployed target platforms. The views are formed from experience of the UK market, which has required an understanding of technologies as diverse as MHEG-5, OpenTV and Liberate.

This document was originally published in the Conference Publication of the International Broadcasting Convention (IBC 2001) Amsterdam, 13–18 September 2001.

Key words: digital television, MPEG, DVB, MHP, MHEG-5, OpenTV, Liberate.

White Papers are distributed freely on request.

Authorisation of the Head of Research is required for publication.

DELIVERING INTERACTIVE DTV SERVICES TO MULTIPLE TARGET PLATFORMS

Dr Richard Bradbury, Dr Richard Cartwright and Toby Steele

BBC Research and Development, UK

ABSTRACT

The mature MPEG specifications for digital video and audio ensure the availability of interoperable equipment that simplifies the task of creating broadcast content in a cost-effective way. In contrast, no single technology has emerged as the common standard for multimedia services. Whilst many content providers have achieved the launch of bespoke multimedia services for the deployed target platforms, authoring multiple times is not a cost-effective long-term solution.

Initiatives such as DVB's MHP and ATSC's DASE are attempting to solve this problem by establishing multimedia standards for DTV. Regardless of their outcome, a legacy of already deployed technologies will exist for delivery platforms that are either unable or unwilling to migrate.

This paper describes the BBC's views on authoring and delivering interactive services to deployed target platforms. The views are formed from experience of the UK market, which has required an understanding of technologies as diverse as MHEG-5, OpenTV and Liberate.

INTRODUCTION

It is the BBC's policy that its services should, as far as possible, be available to all licence fee payers without prejudice. This is a key public service commitment that requires the supply of both linear channels and interactive multimedia services to all of the deployed digital television (DTV) networks in the UK.

To this extent, the BBC was the first broadcaster to launch interactive services on all three major DTV platforms in the UK, as shown in Table 1. The digital platforms, terrestrial (DTT), satellite (DSat) and cable (DCable), have independent network operators that have chosen different middleware platforms for commercial and technical reasons. BBC R&D has established an in-depth understanding of these three middleware platforms through its participation in the development of these services.

Date	Service Launch
Nov 1998	Digital Text DTT
June 2000	Wimbledon DTT
Aug 2000	Digital Text DCable
Mar 2001	Digital Text DSat
June 2001	Multistream Wimbledon DSat

Table 1 – Chronology of key BBC interactive service launches.

If a broadcaster wishes to launch a new interactive service, it is currently necessary to write bespoke application code separately for each target middleware layer: MHEG-5 for DTT, OpenTV for DSat and Liberate DTV Navigator for DCable. Three separate programming teams are required and co-ordinating their efforts involves complex project management.

These systems are not binary or source code compatible because each is based on its own hybrid programming paradigm:

MHEG-5 – An ISO/IEC standard for multimedia application description, implemented by an application embedded in each receiver. Code is an object-based declarative description of an interactive service with application behaviour described by state machines.

OpenTV – A proprietary system from OpenTV Inc. with an API implemented over a virtual machine that is ported to each receiver. Code is a procedural description of an interactive service, supporting some aspects of object-orientation, with an event-driven model of behaviour encouraged by the API.

Liberate DTV Navigator – An HTML-based server and browser[1] system with proprietary extensions to support the specification of DTV interactive behaviour through JavaScript. Code describes interactive services through a hybrid of hypertext marked-up content with implicit link behaviour and procedural, event-driven JavaScript for other application behaviour.

This paper describes research at BBC R&D that aims to reduce significantly the amount of specialist coding effort, and hence time and expense, required for the production of interactive services targeting the UK's deployed DTV platforms. The goal is to provide a cross-platform framework in which a service can be described once and published many times. The next section describes the bounds and scope of the research problem. This is followed by an overview of the requirements for a data model that can generically describe interactive services once and then publish to multiple target platforms. Initial research results are presented.

RESEARCH PROBLEM

So-called *new media* technologies now embrace a wide range of different devices, including mobile phones, personal digital assistants, PCs and interactive televisions. The characteristics of the hardware of each of these types of device are vastly different, as each device has a situated purpose. Physically, each type of device is characterised by different display resolutions and different user input interfaces. For example, a PC has a high resolution display (typically 1000 by 1000 pixels) with a mouse/keyboard and is designed to be viewed close-up on a one-to-one basis. In contrast, an interactive television has a medium screen resolution (720 by 576 pixels) with a remote control and is intended to be viewed by one or more people from a distance of a few metres.

Rather than considering the generic description of interactive services to suit all new media technologies, the research problem described in this paper is bounded to consider DTV platforms only. In this way, any generic description of an interactive service can assume a common resolution and similar remote control device for user interaction. The large viewing distance for DTV platforms forces presentations to use correspondingly large font sizes and, as scrolling can be counter-intuitive, this requires the segmentation of content into screen-sized sections.

DTV is not the same as the Internet and the task of making Internet content look and feel reasonable as a DTV interactive service must address a number of issues. Scrolling through

[1] The server is a special *transcoding* server that reduces the processing load on the browser application through head-end pre-processing for some aspects of the content.

content and navigating between hyperlinks can be difficult using a remote control because it only has *up*, *down*, *left*, *right* and *select* keys for this purpose. For example, it is ambiguous whether pressing the *down* key will take you to the next hyperlink in the page or scroll through the content. To prevent scrolling, the layout specification of a service needs to be explicit, as the DTV receiver does not necessarily implement a presentation browser. In other words, a generic description cannot assume a browser that resolves presentation issues based on marked-up content.

The information to describe an interactive service can be classified into four categories:

content – The content assets to be presented (text, image, graphic) and the editorial selection and ordering of that content.

template – The way that the content should be presented in terms of layout, design and rules for presentation.

navigation and structure – The way that the information is structured and accessed by a user, e.g. page numbering system, information hierarchy, virtual book.

behaviour – The description of the interactivity provided by an application including handling user events, timers and triggers, i.e. how to describe the workings of a menu, quiz, game or live event.

When describing an interactive service, it is not possible to express each of these categories independently: they are inter-related. Consider a long piece of content that is too big to fit within a layout template on a TV screen, as illustrated in Figure 1. In this case, the template needs apply some rules as to how the content should be broken into separate screens. This then affects the behaviour of the application, with the need to add **NEXT** and **PREV**ious buttons to flow content between screens. If an additional **LINK** to further information is included, this may be defined to be at the end of the content and hence the service navigation structure is altered.

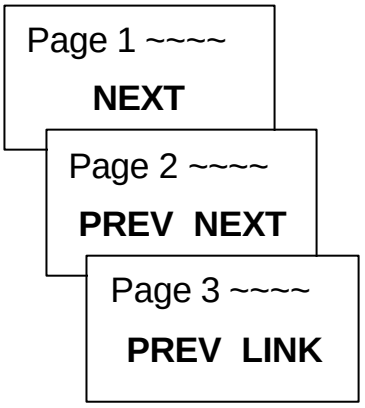

Figure 1 – Binding content, templates, navigation and behaviour

Beyond the scope of the research problem is the implementation of a Content Management and Production System (CPMS) for DTV interactive services. The focus of this work is on the service publication, as illustrated in Figure 2,

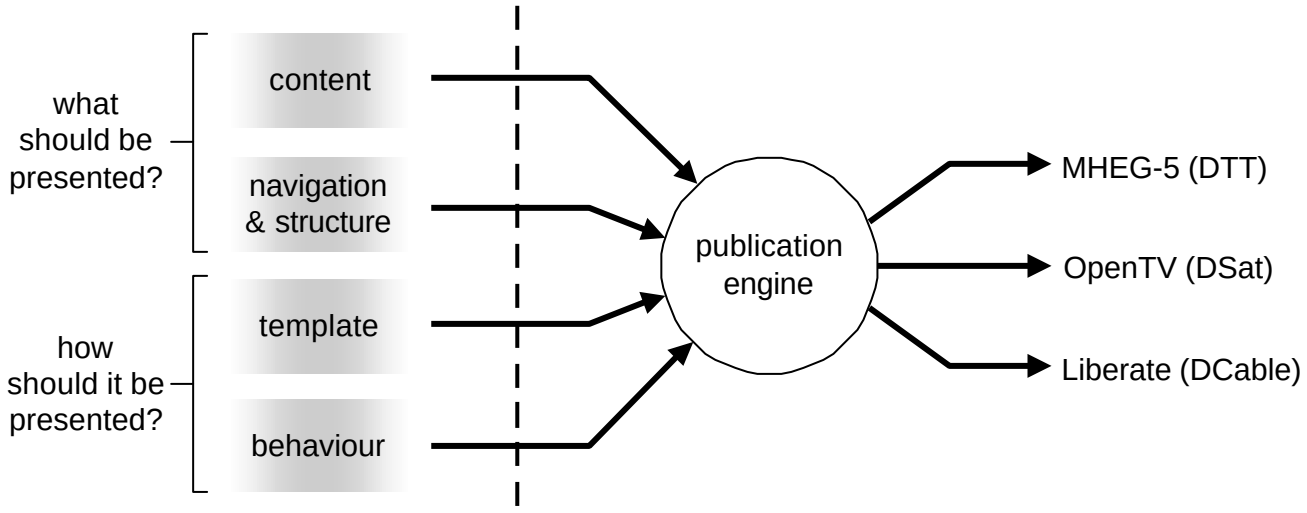

Figure 2 – Interactive service description interface and engine.

where it is assumed that the content is already aggregated into a publishable form. The key part of this is a data model that enables content to be passed across an interface (dotted line) partitioned according to the roles of people engaged in producing the content. On the downstream side of this interface is a *publication engine* that can automatically translate the service description into an appropriate format for each target platform and interface with the various broadcast playout systems.

Table 2 shows four features common to all DTV interactive services and the way they are deployed in the UK for each target platform. These features illustrate aspects of interactive service description that our data model intends to hide through abstraction. It will be a task for the publication engine to resolve the high-level description to the platform specific version.

Feature	DTT – MHEG-5	DSat – OpenTV	DCable - Liberate
Font	Tiresias embedded at sizes 36, 31, 26, 24pt – no downloading. Kerned.	Helvetica embedded and scalable – can download bitmap and soft fonts. Tiled.	Tiresias default on transcoding server – can download from server. Tiled.
Image	PNG, and MPEG I-frame windowed from video layer.	BMP encoded to proprietary run length and MPEG I-frame windowed from video layer.	GIF, PNG or JPEG, encoded to a proprietary 15bpp format on transcoding server.
Palette	Embedded: 139 opaque colours, 48 transparent colours and transparent – content broadcast at 32bpp (RGBA).	Broadcast, with a maximum of 256 colours at any one time – content broadcast by palette index.	15bpp RGB direct colour, described in content at 24bpp. Support for alpha channel only in static GIF.
Remote control	Must meet minimum UK MHEG Engine Profile – otherwise different for all receiver models.	Specified by BSkyB – receiver independent. Lacks DTT *EXIT* key and adds *back-up* key.	Design agreed with network operator. Lacks DTT *EXIT* key, and adds *Back, Page-up*, *Page-down* and *Skip* keys.

Table 2 - Feature comparison between deployed UK DTV platforms.

One possible approach to the resolution of this problem is to implement a browser for each platform that can then receive the same broadcast content. The problem with this approach is managing the different characteristics of each platform and the efficiency of the browser application implemented as an additional level of processing. The common content that is broadcast either has to be tailored to work on the lowest common denominator platform or carefully mangaged in the case that the content type is not supported. The bandwidth for each platform is assumed to be the same, with a fixed browser interface dictating the separation of head-end and receiver processing. The problem with deployed platforms is to find a flexible balance between head-end and receiver processing, tuned to provide high-quality services for each target platform.

DATA MODEL REQUIREMENTS

Public Service Broadcasters produce a diverse range of television programmes and many are suitable for interactive enhancements. To fulfil the requirements to produce interactive services across all creative genres, it is necessary to find a data model for these services that is well defined and can be used by internal and external production teams. In this section, the requirements for the data model and the implications for the implementation of a publication engine for this model are described.

Separation of Production Roles

Different aspects of an interactive service derive from different roles within a production team. The presentation layout of a service is controlled by a designer, the behaviour of the application is agreed with technical editorial staff, content is authored and aggregated by a journalist, and the overall service structure is defined by a producer. The data model should respect and enable the separation of these roles through its abstractions, hiding the need for any individual to have specialist knowledge of the target platforms during the production process. It should be possible for existing CPMSs to write to the data model. Operations staff should be able to modify and configure the service as it is broadcast, also without the need for specialist platform knowledge.

This implies that the data model should permit fragmented description of areas such as presentation template, content and behaviour. The publication engine should be capable of combining these aspects of the model as required and generating the appropriate platform-specific service.

Service Lifecycle

Some services are created once and not subsequently updated, requiring a set of files to be generated that remain the same over the lifetime of the service. Other services undergo periodic changes to appearance, content and/or structure. Yet another class of services integrates near real-time content synchronised with live events or triggered from associated video streams. The data model is required to be able to represent each style of service lifecycle.

To support the different service lifecycle styles, the publication engine should be capable of:

- generating one-off static services offline;
- connecting to online broadcast systems and managing dynamic service change.

In the case of near real-time content, the system should be able to generate service updates within a specified tolerance and without the need to rebuild the entire interactive service. For the data model, this implies that it should be possible to post updates through the system in small fragments without the need to send the entire service description.

Abstraction from Platform Specifics

The generic model of an interactive service should be a high-level abstraction from any DTV platform-specific knowledge required to code interactive services. Although the data model itself may only be understood by interactive service specialists, it should be possible to implement graphical tools for members of production teams to carry out their roles without the need for any in-depth technical knowledge. For example, a designer could be provided with a drag-and-drop design tool, configured to assist with design for the TV screen and an output to the generic data model. This abstraction should admit extension to new forms of presentation

and interaction as required.

This requirement has significant implications for the publication engine. It must be able to perform a translation to platform-specific targets without intervention from the content creators, including text encoding and image conversion. To achieve this, it will be necessary to describe services in terms of nested interactive service *components* in the data model. If an implementation of a component for a particular target platform is available, it should be recognised from the service's data model by the publication engine. Components and the means for their specific instantiation will initially have to be coded for each platform by a specialist, then establishing a toolkit of reliable components available to all future applications.

The DTT and DSat platforms deployed in the UK use modular carousels to broadcast the application and content data for interactive services to receivers. (DCable is migrating towards this model.) The perceived speed of the interactive viewing experience is crucially dependent on the configuration of these carousels. Currently, services are partitioned statically for each platform according to rules of thumb. To achieve complete abstraction, this process will require automation within the publication engine.

Platform-specific Optimisation

Each platform will have its own technical strengths and weaknesses. The available bandwidth for an interactive service is known from a process of carefully balancing a platform's total bitrate budget with the requirements of all its services. This can vary significantly from platform to platform. The data model should abstract a non-specialist user from the need to understand the detailed specifics of the uniqueness of the platforms. At the same time, technical editorial decisions should be supported to enable flexible tailoring of applications to suit particular platforms and their bandwidth.

In implementing this flexibility, it will be necessary to optimise the code generated for each presentation component to suit its platform. This will often involve balancing processing between the head-end and receiver for each component on each platform. For example, either the head-end or the receiver may take care of rendering marked-up text according to the processing power available in the receiver. Some automatic identification of repetitive use of an asset in multiple scenes may be employed to eliminate repetition in the broadcast carousels. In the case where services diverge in some way across multiple platforms, perhaps due to their bandwidth limitations, a broker should be implemented to manage divergent service changes.

INITIAL RESULTS

An initial data model has been defined and is the basis for an experimental framework with which to test the feasibility of new ideas. Models can be expressed according to an XML Document Type Definition (DTD), which acts as a convenient way to package up service descriptions as files or network transactions. This notation has a clear separation between content and design, enabling the service to be described in a purely abstract way that respects the roles of members of the production team.

A prototype implementation of a publication engine that supports the description syntax has been developed at BBC R&D. Developed using the Java programming language for portability, the publication engine manages the dependencies that are established between content and templates. To automatically generate platform-specific versions of services, content and templates are bound together through instantiation of platform-specific

components. Updates can be posted in fragments to the system during the lifecycle of the application and the engine ensures that all related aspects of the service are updated.

Design templates are expressed in terms of abstract presentation components parameterised by properties such as position, size and colour scheme. Simple components, such as text, bitmap and rectangle, map directly to presentable objects on the receivers. Composite components, such as menus and push buttons, are assembled from collections of simple ones under the control of an automatically generated state-machine describing their behaviour. For example, a menu is assembled from a collection of text boxes and a state machine. This state machine embodies the behaviour associated with the up and down keys used to highlight the currently focused item.

Internally, the publication engine has classes directly representing the objects available for each platform and can generate the code representing instances of these objects on request. The publication engine maps the abstract description of the service to collections of instances of these classes, automatically handling issues of text encoding, image conversion and the resolution of composite components into platform specific collections. New composite components can be added on-the-fly. In this way, the author of a description in the DTD syntax does not need to know how to code for a particular platform and instead relies on a library of pre-built components that are created by a specialist.

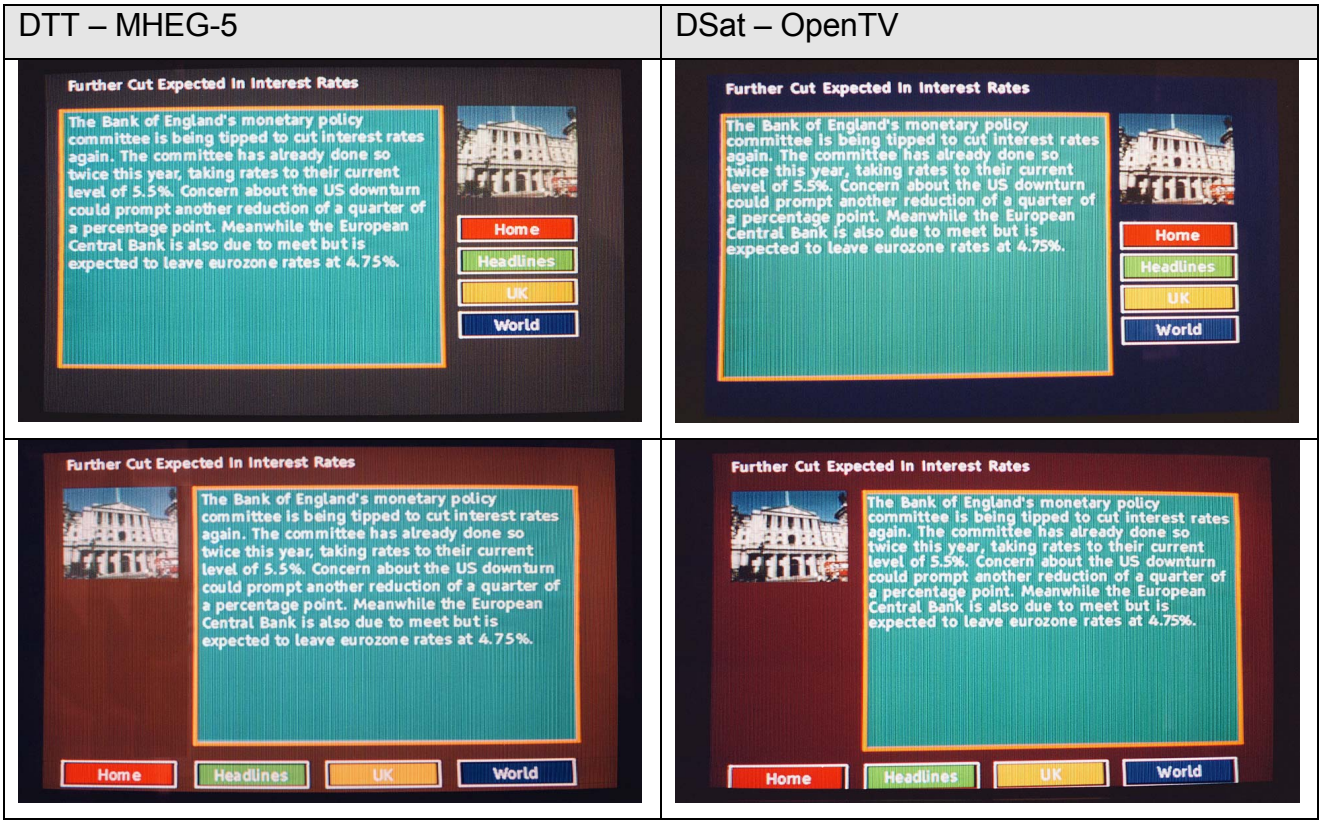

Figure 3 – Comparing publication engine output for DTT and DSat.

A particular instance of the engine is configured to produce a nominated platform's output, although input is the same in all cases. Using an HTTP server accepting requests for changes, it is possible to post updates to different platforms either synchronously or independently, allowing the services on different platforms to diverge where necessary. Currently, the most mature output is MHEG-5, with direct generation of ASN.1 encoded MHEG-5. The OpenTV output has comparable functionality but is less optimal due to the

overhead of compiling to the proprietary *o-code* required by the receiver's virtual machine. An HTML/JavaScript output for use with the Liberate DTV Navigator platform is still in development.

Figure 3 shows output of the publication engine from one common interactive service description for a *digital text* style case study. The top two images show a page in the service with the initial template and the bottom two shows the service after a template change has been posted to the publication engine, causing the layout of the page to be altered. Without close inspection, it is very difficult to see the difference between the DTT and DSat versions and this is a successful result. In Figure 3 it is possible to observe a difference in line spacing for the wrapped text. It is less obvious that the colour palettes differ and the publication engine has dithered the images differently as a result.

CONCLUSIONS

This paper presents initial research concerning the generic description of multimedia content for multiple target DTV platforms. The key deliverable of this research is a data model that supports the separation of production roles:

editorial - description of services in a flexible and platform-independent manner;

technical editorial - utilisation of the capabilities of each target platform, supported through componentisation;

operational - support for dynamic configuration according to operational requirements;

specialist technical - flexibility to distribute processing between head-end and receiver, tailored to the capabilities of each platform.

A prototype tool, known as a *publication engine*, supports the development of the data model. This has allowed us to validate the emerging model, to ensure that it is feasible to implement and that it meets real-world requirements.

ACKNOWLEDGEMENTS

The authors would like to thank their colleagues for their contributions to this work. In particular, Jeff Hunter has provided support and encouragement to the team and Johann Broomfield has contributed significantly to the development work. The authors would also like to thank the British Broadcasting Corporation for permission to publish this paper.