Semi-automated Mobile TV Service Generation

Dr Moxian Liu, *Member, IEEE*, Dr Emmanuel Tsekleves, *Member, IEEE*, and Prof. John P. Cosmas, Senior *Member, IEEE*

***Abstract*—Mobile Digital TV (MDTV), the hybrid of Digital Television (DTV) and mobile devices (such as mobile phones), has introduced a new way for people to watch DTV and has brought new opportunities for development in the DTV industry. Nowadays, the development of the next generation MDTV service has progressed in terms of both hardware layers and software, with interactive services/applications becoming one of the future MDTV service trends. However, current MDTV interactive services still lack in terms of attracting the consumers and the service creation and implementation process relies too much on commercial solutions, resulting in most parts of the process being proprietary. In addition, this has increased the technical demands for developers as well as has increased substantially the cost of producing and maintaining MDTV services. In light of the aforementioned situation, this paper proposes an innovative MDTV service creation and consumption system based on XHTML and Java ME. On the head-end it introduces a semi-automatic creation mechanism to facilitate a less technical and more efficient interactive service creation process. This enables designers and creative individuals to be actively involved in the MDTV service creation process and to develop interactive-rich MDTV service. On the client-end it employs an open-source software environment as the interactive service MDTV consumption platform, rendering the MDTV service implementation process as less proprietary as possible. Initial tests indicate that the proposed solution allows for a faster and more effective MDTV interactive service generation.**

I. INTRODUCTION

Modern Digital TV (DTV) systems broadcast not only traditional Television (TV) content but also provide viewers with a series of interactive TV services. Following the dramatic increase in the use of personal mobile devices such as mobile phones, PDAs (Personal Digital Assistants), notebooks and netbooks, DTV has today been realized and ported in the mobile environment creating the advent of Mobile Digital TV (MDTV).

The majority of current MDTV related research work and commercial solutions mainly focus on shifting existing DTV services to MDTV, even though the characteristics between DTV and MDTV systems vary considerably. More precisely, research has been conducted in recent years on constructing the MDTV service distribution (broadcasting) and reception environments from both hardware and software perspectives. The development results on the hardware environment and lower layer protocols are promising with reliable solutions (broadcast networks such as DVB-H and mobile convergence networks such as 3G) being formed. However the implementation of higher layer components is running relatively behind with several fundamental technologies (specifications, protocols, middleware and software) still being under development.

The vast majority of current commercial MDTV service applications include free-to-air, Pay Television (PayTV) and Video-on-Demand (VoD) services. When contrasted with the various service types of conventional DTV, MDTV services are still unidirectional, offering basic services that lack in attracting large numbers of subscribers. Services with more interactive features found on DTV systems (such as multiplayer gaming, voting, and online betting, etc.) are required by users on the MDTV platform too. Therefore under the provision of mobile telecommunication network as the interaction channel, discussions and attempts at introducing interactive-rich services into the MDTV domain have been ongoing. However, various proprietary commercial solutions dominate the current interactive service generation and implementation field. This has resulted in a plethora of fragmented proprietary middleware/software specifications. The lack of a universal or an open-standard solution constitutes one of the current issues limiting the speed of evolution and richness of MDTV services, especially in terms of the deployment of interactive applications [1].

This paper presents a novel solution for the creation of interactive MDTV services and applications, with a twofold aim. Firstly to introduce open-standard technologies in the service creation process, in order to improve the compatibility of MDTV services with different bearer standards; and secondly to reduce the technical demand on the service creation process, allowing designers to be actively engaged in the service creation process. The paper is divided into five sections. Section I provides a brief overview of the MDTV service development. Section II provides an insight into various service generation solutions in the current MDTV industry and research field. Section III presents the proposed solution including its adopted technologies, assistant software tool, the data flow throughout the service generation process and a series of comparisons with other solutions. The initial test results and a discussion of the proposed solution are provided in Section IV. Lastly, section V presents the future work and the conclusions.

Manuscript submitted March 31st, 2011.

M. Liu, E. Tsekleves, and J. P. Cosmas are with Brunel University, London UB8 3PH, U.K. (e-mail: moxianliu@gmail.com, Emmanuel.tsekleves@brunel.ac.uk, john.cosmas@brunel.ac.uk, Telephone No.: 01895-267379, 01895-266756).

II. Mobile Digital TV Service Generation

A. MDTV service generation and implementation process

The mobile version of DTV, MDTV shares similar service generation and implementation process with it. A typical process chain is illustrated in Fig 1, where there are five key entities: the Content Provider (CP), the MDTV Service Provider (MSP), the Broadcast Network Operator (BNO), the Cellular Network Operator (CNO), and the Terminal Device Manufacturer (TDM). The BNO and the CNO are the bearer network providers. The BNO provides the broadcasting network resource for the service distribution whilst the CNO provides the interaction channel for the service consumption [3][4].

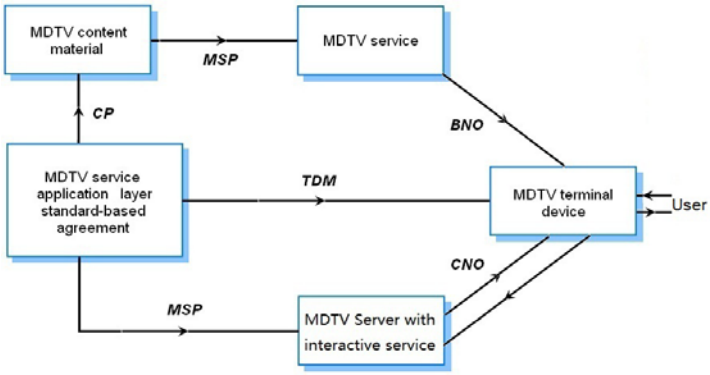

Fig 1: Typical MDTV service implementation process chain

The key component in the chain is the MDTV service application layer agreement. This agreement usually consists of the MDTV middleware specification definitions such as the service content (audio, video, and other auxiliary data) formats, service application model, application execution platform, application signaling, transport protocols, graphics model and security mechanism. Based on the bearer MDTV network standard, the agreement is arranged either according to the pre-defined middleware/software specification of the network standard or through a series of development work conducted by the relevant commercial entities. The CP, MSP and TDM take this agreement as the reference during their work within the chain. For instance in a DVB-H based MDTV service implementation, the bearer network standard is DVB-H and thus the application layer agreement will employ DVB-IPDC as the basis since it defines most of the required middleware components. For those required components, which are not fully defined in DVB-IPDC, such as the service application model and the application execution platform, some development work will be conducted based on the requirements from the relevant CP, MSP or TDM. Once all the required specifications are defined and accepted by the CP, MSP and TDM within the service implementation chain, the application layer agreement is then formed and is followed during the actual DVB-H based MDTV service implementation work.

The CP is usually responsible for collecting MDTV service content such as recording TV programmes, providing assistant information and completing the initial productions of the TV programmes. The MSP is responsible for collecting MDTV content from various CPs and incorporating them into MDTV services, by using a number of appropriate service authoring tools. These are then distributed to the end-user with the help of the BNO. In addition to this, the MSP is responsible for setting up the server-side content for the MDTV services it provides and cooperates with the CPs to actualize the implementation solutions for the interactive applications. The TDM has to follow the agreement and provide users with the MDTV terminal devices that are able to meet the MDTV service requirements within the chain; such as capable of receiving services from BNO, capable of retrieving the services provided by MSP, capable of running any interactive applications by cooperating with the CNO and MSP.

When a MDTV service is generated by the CP and MSP and then distributed by the BNO, the user (consumer) can start consuming the service on the MDTV terminal device produced by the TDM. The consumer has the option of either watching the produced MDTV content or interacting with the interactive components of the service. Such interactions are accomplished through the interaction channel provided by the CNO, where the consumer requests (such as a VoD command) are sent from the device to the server and the server processes and responds to them.

B. Current service generation overview

Based on the aforementioned typical service generation and implementation chain, the relevant commercial entities can successfully deploy their MDTV services in the current market. With regards to service generation, there are mainly two different solutions being adopted, namely the standard-based solution and the commercial solution. The difference between these emanates from the discrepancies in their corresponding service application layer agreement.

1) Standard-based generation solution

There are MDTV service generation solutions that apply the middleware and software specifications of their bearer network standards as the service application layer agreement. These solutions are defined here as the “standard-based generation solutions”. In practice, this type of service generation solution is usually preferred by ATSC-M/H, ISDB-T 1seg, DMB or 3G network based MDTV service implementations since all these standards have their own middleware or software specifications such as the ATSC-M/H Application Framework[23], ISDB-BML[24], DMB-MATE[25][26], 3GPP-DIMS[27] and OMA-RME[28][29]. In this type of solution, the various commercial entities (e.g. the CP, MSP and TDM in Fig 1) have minimal involvement in any technical development work and most of their work is based on the execution of those specifications as required.

The advantage of such standard-based solutions is that the service generation is relatively pre-defined and thus no additional resources (time or cost) is spent on developing or deciding the technical components defined in a middleware/software specification. However, these solutions do not encourage the wide involvement of designers in the service creation process. This is because considerable technical training is required with regards to the technically challenging corresponding authoring tools developed for generating the

service content. More importantly, those standardized specifications are usually compliant only to their corresponding MDTV network standards, thus the services generated are restricted in terms of wider deployment. Therefore the relevant commercial entities often have to adopt different schemes in order to create/implement the same service (such as a TV programme service) for each of its possible underlying MDTV network standards. This leads to an increase of the time, costs, as well as staff resources required for the production of an MDTV service.

2) Commercial generation solution

Apart from the aforementioned standard-based solution, any solution that relies mostly on the commercial development during the definition of its service application layer agreement is defined here as the "commercial generation solution". Such solutions are utilised mostly for a quicker deployment but they ensure a relatively normative implementation as standard-based solution does. In practice, a commercial solution's application layer agreement usually defines a similar software mechanism with that of the middleware/software specifications but with most of the features developed against the related service requirements. Commercial entities such as CP, MSP and TDM can be involved in such agreement development work according to their different development requirements.

Several of the commercial MDTV service generation solutions [20]-[22], [30]-[33] are more customized against the relative service requirement when compared to standard-based solutions. More importantly, they usually support multiple standards, resolving the compatibility problem found in the standard-based solution. Such solutions however raise proprietary issues. Corresponding commercial entities own the copyrights of the products as well as the relative technologies and the maintenance/update operations have to be under the entity's authority. This proprietary character may hinder a commercial solution from achieving wider industry support and adoption. Moreover, a proprietary solution usually requires individuals with more technical expertise to develop MDTV services using customized and specialized tools. Since for working with those tools along with the relative proprietary specifications and protocols defined in the solution, designers usually need additional knowledge learning and skill training before their actual work. This thus increases the difficulty and the complexity during the maintenance and usage of the solution.

Moreover, regardless of the different types of current MDTV service creation solutions, a common problem that contributes to this situation has been identified as the high technical demands on the designers during the service creation process. Typically a MDTV service designer is required to have very specialised technical knowledge and skills especially in the following three different areas. Technical knowledge and skills for software development on mobile device, design and graphics skills on developing web-like content and applications, and technical knowledge and skills on developing MDTV services/applications based on different MDTV standards. As a result the requirement of such a high demand of background knowledge and skills hinders creative and professional designers from becoming involved in the MDTV service creation process, requiring a considerable amount of training. This situation contributes further to the lack of design and creative related individuals in the MDTV service creation field, hindering in turn the development of visually appealing and engaging MDTV services.

C. Universal service generation solution - Related work

Since both of these types of solutions have a number of issues, there is a requirement for a universal solution that could benefit from all the advantages of the two solutions and resolve their underlying problems. However due to the complexity of developing such universal solution, there are only a few related research pieces of work from the academic research field. Two types of methodologies have been proposed to that effect.

One is based on a recommendation to improve the inter-compatibility of the output service content and applications, where there should be a uniform middleware/software architecture for different network standards as the service generation reference. More precisely in one of the related research projects, an interoperable middleware architecture is proposed for digital broadcasting [5]. This architecture offers a more flexible management of middleware from different conventional DTV standards (e.g. DVB-MHP, ATSC DTV Application Software Environment, ARIB-BML, etc.) and is able to execute the various middleware applications that are developed for a specific standard. This means that any DTV service generation solution that employs this architecture as the basis of its application layer agreement is able to ensure the inter-compatibility of its output service to multiple standards. Even though this research work is related to conventional DTV service, it provides a valuable reference for the paper's research in that by integrating multiple MDTV middleware/software specifications, the existing incompatibility problem could be resolved and the standard-based service generation solution can in turn be improved to become a universal solution.

The other methodology suggests the introduction of an open-standard technology and an open-source development strategy in the commercial service generation solutions. Both of these two concepts have already contributed to the current IT field in many aspects, such as encouraging the development of software products (e.g. XHTML based web services and Java based software applications, etc.). However in the digital broadcasting service generation field, there are a few successful adoptions of these two concepts, principally due to the complexity of the development work. However there is one related research work in which [6] after a comparison between several existing middleware platforms of digital TV including MHP, DASE, BML and MIDP, it concludes that Java ME MIDP has several advantages and can be used to fulfill the requirements of an MDTV services. Such an open standard software system can be introduced in the current MDTV service generation. Together with its excellent functionality and adaptability, the quality of service could be improved and more importantly, proprietary issues would be prevented during the commercial implementations. Java ME MIDP is therefore a suitable candidate technology for the development of universal MDTV service generation solution.

III. Semi-automated MDTV service Generation

Motivated by the requirement of developing a universal solution and further inspired by the related work, a novel universal solution is proposed in this paper to allow for semi-automatic MDTV service generation. Such solution consists of an assistant service creation software tool and a new service presentation method.

Fig 2 illustrates how the assistant software tool is integrated in the current MDTV service production flow. In the original service creation process, the MDTV content (including audio, video, images and texts, etc.) have to be produced by the multimedia service provider (MSP) in order to become MDTV services ready for broadcasting. In contrast in the proposed semi-automated service generation process, a service creation tool is introduced, which can be utilized by the content provider (CP) to produce the MDTV services. With the assistance of the proposed toolkit, the CP is able to complete the whole service generation process without support from the MSP. The original process is thus facilitated so that considerable amount of collaborative work between CP and MSP can be saved. Moreover, the proposed tool offers a semi-automatic MDTV service creation mechanism where the tool handles most of the complicated and technical elements of the MDTV service generation automatically; the designer can focuse on the service design aspects and apply the required MDTV service functionality via the use of drag-and-drop functions. As a result of this, there is a lesser demand on the designers to have advanced technical skills, further facilitating the MDTV service production. In addition to this more creative designers from other IT and design fields can be encouraged to participate in the service generation process.

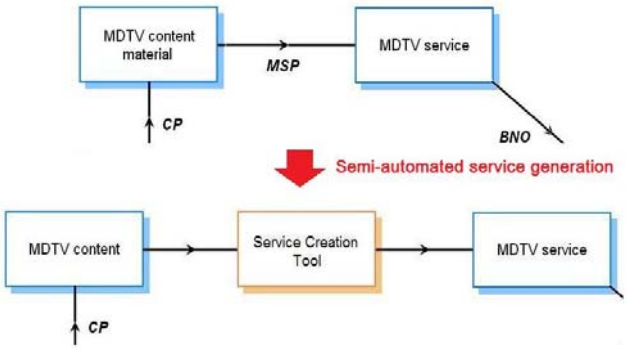

Fig 2: Integration of proposed tool to the current service production flow

As mentioned in Section II A), the relevant entities (CP, MSP and TDM) implement the MDTV service according to the predefined service application layer agreement. When producing the service, the CP and the MSP usually take two definitions of the agreement, namely the content format and the application model, as the service presentation reference. As a result, various MDTV service generation solutions have different service presentation methods regarding to their design requirements. The proposed service presentation method is thus developed aiming at providing reference for the proposed semi-automated MDTV service generation solution. The method adopts open-standard technologies including eXtensible Hypertext Markup Language (XHTML) and Java ME and provides the same or better functional supports to the multimedia and interactive-rich MDTV service than other methods currently used. This will be discussed in detail in the next section. In a nutshell, the XHTML provides as good multimedia presentation functions as XML and Rich Media but a much easier development and better compatibility to different screen sizes and resolutions; Java ME provides more powerful functionalities and better adaptability than EMCAScript when presenting the interactive applications on mobile devices. Moreover this novel method helps in minimizing proprietary issues during the service generation and lowers the technical demands since both XHTML and Java ME are mature technologies that many professionals have a wide experience on and several authoring tools are available in the IT field. More importantly, the proposed method unifies the service presentation methods of most of the current MDTV standards to ensure its corresponding services are compatible with various service systems and platforms.

Taking the new service presentation method as a reference and with the assistance of the service creation tool, the proposed semi-automated MDTV service generation solution can then be implemented. The milestones during the introduction of the proposed solution are recognized as the following.

A. Better presentation technologies selection

Fig 3: Typical MDTV service presentation

A typical MDTV service (see Fig 3) consists of Audio/Video data and auxiliary data services, whereas the auxiliary data service includes content in multiple formats such as text, image, animation, and interactive applications like voting, gaming, or chatting. These are created against different service requirements which are the audio-visual components visual presentation technologies and the interaction presentation technologies. As illustrated in Fig 3, the visual presentation technologies are applied to format and present all the visual components of the MDTV service whilst the interaction presentation technologies enable the running of the interactive components and applications of the MDTV service and enable interaction between the service and the user.

Amongst the different visual presentation technologies applied in the current MDTV industry such as XML, Rich Media technologies and XHTML. The later has been chosen as the visual presentation technology for the proposed service

presentation method. The considerations behind this are presented below:

The Extensible Mark-up Language (XML) [36] is designed to transport and store data rather than designed for displaying data like XHTML. In order to display the content in a XML format, there is always need of a specially designed presentation engine to assist in parsing and rendering the data. Examples of such presentation languages include the XML User Interface Language (XUL) from Mozilla [7] and User Interface Markup Language (UIML) from Oasis [8]. Those versions of XML are usually customized by the developer for specific requirements. In turn, the authoring tool ~~as well as~~and the rendering engine on the client side have to be designed for each specific customized XML description language. This results in that those specific solutions cannot be accepted universally by different bearer standards and they require extra resources for implementation across different terminal platforms.

Rich Media technologies are widely adopted as the visual presentation technology in many current MDTV standards because of their excellent graphic and animation presentation ability. Scalable Vector Graphics (SVG) [34] is a typical one of them and its sub-version know as SVG-Tiny has been used to define relative specification in standards such as 3GPP, ISDB 1seg and ATSC-M/H. Commercial solutions such as Streamezzo also apply it for their visual presentation. However, the Rich Media featured MDTV service implementation suffers from several drawbacks.

Firstly, rich Media based service contents/applications can only fit the specific screen size of a MDTV terminal device that they were originally designed for. This implies that a MDTV service has to propose different Rich Media UI solutions regarding the different types of terminal device screen sizes and resolutions. This may lead in additional development work on UI design and service layout design and in turn result in the increase of project costs and the complexity of system maintenance and update.

Secondly, most of the Rich media technologies require advanced technical knowledge. Flash Lite [35] is a proprietary Rich Media format that requires the end-to-end proprietary development and deployment including the underlying technology, authoring tools, server and the Adobe Flash Player client. Although SVG can be edited in a plain text-editor, the corresponding authoring tools are usually necessary for an efficient and effective design. However, three main sets of Rich Media technologies, MPEG4 BIFS [18], MPEG LASeR [17] and SVG Tiny, which are currently used in the MDTV service development, offer different graphic features and in turn they are incompatible and require different sets of authoring tools [9]. Specialized professional knowledge is thus needed when using either of them. Moreover there are no promising Rich media authoring tools suitable for MDTV service and MDTV UI design and most of rich media based service development still depends on proprietary and customized tools.

Moreover, its advanced technical requirements encourage more proprietary commercial solutions dominating the current MDTV service generation field. Therefore regarding the current research scope, Rich Media is not suitable for the development of the universal service generation solution.

XHTML [37] is a sophisticated presentation language widely used for web service development. As to the MDTV service industry field, many leading MDTV standards such as 3G, ISDB 1seg, and DMB have employed XHTML as an optional element for their visual presentation specification. Comparing with Rich Media, the XHTML based MDTV service content have similar characteristics with the classic web services. Along with the web-browser-like terminal service platform, an XHTML MDTV service's UI and layout is more flexible to fit in any device screen size with different resolutions. This means that as long as the service platform is compatible with different sizes, the MDTV service that is in XHTML format is relatively flexible with regards to the layout design. This advantage saves time and cost in a MDTV service developing life-cycle and simplifies service maintenance.

Due to the popularization of XHTML technology in the IT field, the selection of an authoring tool for XHTML is also much easier when compared to Rich Media. Software such as Microsoft Expression Web and Adobe Dreamweaver, as well as several other freeware tools offer sophisticated SDKs for XHTML based design. Moreover, designers and developers that are experienced and active in web service design can be attracted and involved in the MDTV service design. Thus this enables large numbers of designers in the web design field to design services for MDTV as well. Besides, with the help of the proposed assistant service creation tool, the proposed service generation solution can be standard-independent and service-independent.

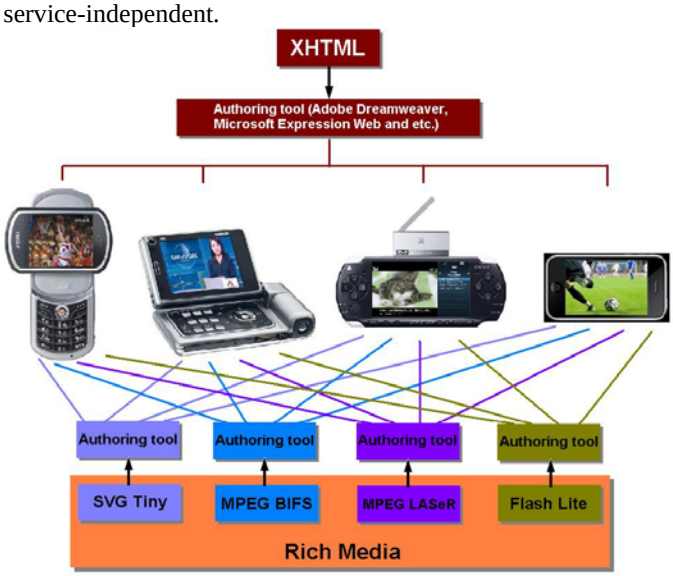

Fig 4: Comparison of authoring tool between XHTML and Rich Media

As to the interaction presentation technology, ECMAScript is selected by most of the DTV and MDTV standards in their corresponding application layer specifications (e.g. DVB-GEM [38], 3GPP-DIMS, OMA-RME, DMB-MATE, OMA-ESMP [39] and HbbTV [40]), although in the proposed presentation method Java ME is chosen instead.

ECMAScript is usually adopted to assist visual presentation technologies like XML, SVG and XHTML in order to provide those markup languages with the following capabilities:

- The ability to apply mathematic and procedural logic locally to document data;
- Providing access to facilities of the device such as messaging function on a phone;
- The ability to generate messages and dialogs locally, reducing the need for expensive round-trip for alerts, error messages, confirmations etc;
- The ability to handle events;
- The ability to allow the dynamic creation and/or modification of documents on the client [10];

A problem though exists during the adoption of ECMAScript is that a scripting language is involved as the interaction presentation technology. Thus its corresponding scripting code parsing mechanism is needed in the terminal device service platform. This translates into additional resources including processing threads, memory and power that are going to be required. This approach is against the common software development strategy on the limited capability of a mobile device. Alternatively, with the excellent native compatibility and supported by many industrial mobile device manufacturers such as Nokia and several other, Java ME can be an ideal choice other than ECMAScript where all these functions mentioned above can be handled by programs developed using the MIDP (JSR118) API [41]. Besides the core specification API in MIDP package, JCP have been developing more assistant optional API packages for MIDP, relevant to the requirements and improvements of the MID's functions, such as JSR82 for Bluetooth, JSR172 as J2ME web services specification, JSR209 for advance graphics and user interface, JSR927 for Java TV API, JSR280 for XML API and many more.

B. Assistant tool and semi-automated service generation process

Although XHTML and Java ME can help towards the development of an open-source solution, knowledge and skills related to XHTML and Java ME are still required on the designers' end including additional programming effort during the service generation. An assistant software toolkit may be required to facilitate this. On top of that, most of MDTV standards have defined their own metadata formats and functional APIs (for example: the URI for audio/video streaming) in their relative middleware and software specifications for service generation and implementation. This means that two MDTV services based on different standards, even if their service creation solutions apply the same presentation technologies and methods, their corresponding metadata and relative functional API would be different. Therefore during the service generation, besides the requirement of some assistant toolkit with a functionality for handling XHTML and Java ME based manipulation, additional functionality is also needed for editing the relevant metadata or functional code in the XHTML source code for various service applications (e.g. interactive application) according to the design requirements.

The aforementioned factors have motivated the introduction of the semi-automatic creation tool, which is the core software component in the proposed semi-automatic service generation solution. By using this tool, service designers are allowed to further manipulate the XHTML pages, make modifications and also add necessary MDTV features (e.g. metadata, functional commands or Java ME based interactive applications) to the pages, and finally generate the integrated MDTV service pages more easily without the need to do any programming themselves.

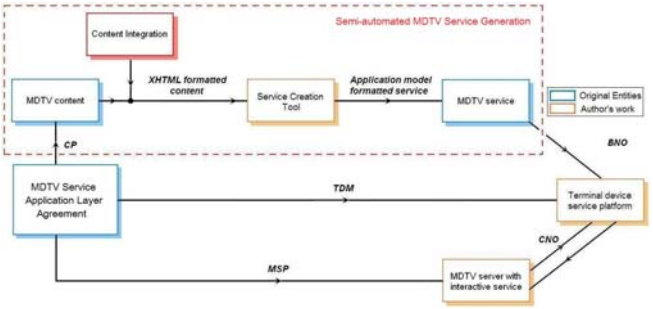

Fig 5: Semi-automated service generation

In the MDTV service implementation entity chain, the MDTV content creation is usually done by the Content Provider (CP), and the service creation is usually done by the Multimedia Service Provider (MSP) (refer to Section II A). By utilizing the proposed semi-automatic service creation tool, these two creation processes can be merged into one aggregation as illustrated in Fig 5. In this case, the MDTV content/service creation may be performed by only the CP. More precisely, after the MDTV content have been collected and produced by the CP, the designers in the content integration segment would firstly format these contents into XHTML code. This process is relatively open to web and service designers, enabling them to easily design service UIs that include text, graphics, links and layout information on popular commercial and professional XHTML authoring tools, such as Dreamweaver. A service designer from CP can then use the proposed assistant tool to import and further modify the produced UIs and service layouts, as well as add any required MDTV features (URIs, interactive applications, and service metadata for ESG) into the XHTML code to eventually transform them into a MDTV service ready for implementation.

Therefore, less entities and resources would need to be involved since CP is able to accomplish the entire service generation process without the heavy involvement of the MSP. Moreover, without the need of in-depth MDTV or software engineering background, a service designer can add from a library of predefined applications, actions related to the designed MDTV interactive service on any element on the UI in a drag-and-drop manner. Thus instead of reprogramming from scratch every time a new application/service has to be created, the service creation tool helps designers and creative professionals to construct or modify MDTV interactive services semi-automatically. More effort can be taken from code writing and function development, to be placed on the visual UI design instead. This eventually results in a faster and more effective service development lifecycle.

The main functions of this tool are: general XHTML based authoring; interactive application (including hyperlinks and Java ME based interactive applications) creation/configuration; automatic generation of service application metadata (This

function is currently unavailable and treated as a part of future work). The data flow through the creation tool during the process is as illustrated in Fig 6.

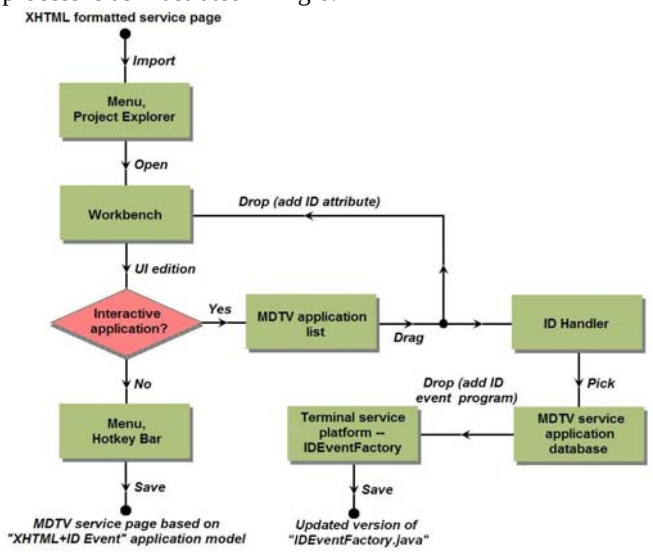

Fig 6: Data flow through the interactive service creation tool

After the MDTV content has been integrated (see Fig 6) in the XHTML format, all the service pages are imported to the service creation tool for further design. The service designer can select those pages through the menu to open them. Both the page preview and the source code are then displayed in the workbench. The designer can carry out a series of design work such as general UI editing. If there are not any additional interactive applications that need to be added in the service page, the designer can save the service page and finish editing. If additional applications need to be added in the page, the designer can swap the workbench to the "element mode" to start the "ID Event" (this is presented in the next section) edit process. The designer first selects the required application from the MDTV application list, and then drags it towards to the target component displayed in the workbench. As soon as the designer drops the selected application, the software requires some relevant parameter configuration according to the application needs (e.g. video resolution, filename, etc). After setting the parameters, two actions are performed automatically: a special ID is added into the target component as a new attribute (ID e.g.: shrek@@cmd0n@@Playvideo@@cmd0v @@root1/320-240(1).mpg@$c0cmd$@. (This is defined in the next section). The workbench updates its scene to display the modifications made to the service page. The ID Handler is triggered and picks up the corresponding application program from the MDTV service application database, according to the name of the dragged application. Then the ID Handler adds the corresponding application program into a special functional class in the terminal device service platform to generate a new version. Finally, by saving the project, the integrated MDTV service page is outputted ready for further processing such as service content packaging. An updated version of the function class for the terminal service platform is outputted as well.

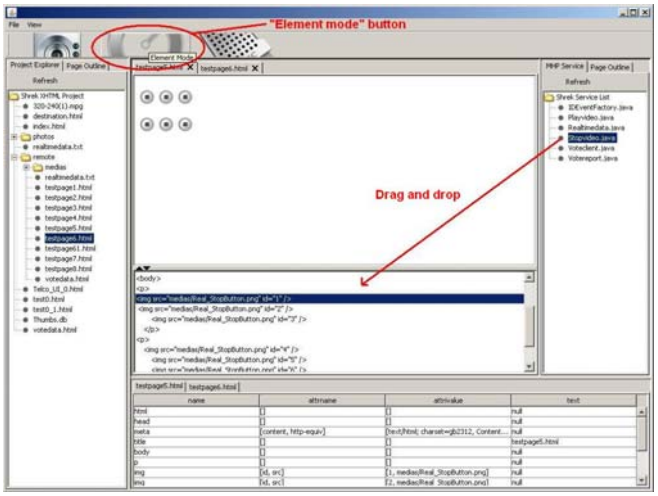

Fig 7: Interactive application configuration to a sample service page

Interactive service applications that are currently supported in the experimental stage within the proposed service creation environment are listed in Table 1.

TABLE 1: SUPPORTED INTERACTIVE APPLICATIONS

Applications	Functions	Descriptions
Media Control	Play media	Start playing the media streamed from a remote broadcast server or local directory.
	Stop media	Stop playing the media.
Live data feed	Require data	Provides live data feed from a remote server and displays it on screen.
Live voting	Vote action	Allows the user to interact with a customized voting application that could be related to the main TV content through a server.
	Require result	Require overall vote result to a specified service component from remote server.

C. Presentation method based on XHTML and Java ME

During the semi-automatic service generation, typical editing work such as XHTML page manipulation or adding MDTV service related metadata into the page can be performed by using basic functions in the proposed creation tool. This type of editing is based on basic XHTML knowledge. On the other hand, when there are requirements for adding interactive applications to the UI components, within the XHTML service page, the manipulations are based on a new interaction presentation method called the "ID Event".

The "ID Event" method is developed as an alternative solution to ECMAScript. This method is enabled by the Java ME MIDP and a special interactive event ID. By using MIDP as well as the required assisting APIs, the service platform in the MDTV terminal device can handle most of the functionalities that scripting languages can provide. By using the special interactive event ID, we bridge the gap between MIDP and MDTV interactive application events, where the service platform can handle the events by passing them to the corresponding event handlers developed in MIDP according to

their ID value. The event ID is integrated into the XHTML code by adding a special ID attribute rather than requiring professionals to manually code it using scripting languages.

The ID in the "ID Event" method is in the form of a string and its format is defined as follows:

Private ID@@cmd**Application serial No.**n@@**Application title**@@cmd**Parameter serial No.**v@@**Parameter value**@$c**Application serial No.**cmd$@

The "**Private ID**" is an identity for information purposes. For example, it can be "DVB-H" to indicate that this ID is designed for service over DVB-H network. Another example is that it can be "John" to indicate the author.

The "**Application serial No.**" is used to separate different applications when there are multiple applications in the ID string. It means that multiple applications along with their parameters can be placed in a numeric order in the ID string. Example:

xxxx@@cmd**0**n@@Application 1 title@@ cmd0v @@ Parameter value @$c**0**cmd$@@ cmd**1**n@@Application 2 title@@cmd0v@@Parameter value@$c**1**cmd$@.....

The "**Application title**" is automatically generated by the proposed service creation tool during the "drag and drop" function of the application according to the name of the selected application in the application database.

The "**Parameter serial No.**" has a similar function with the "Application serial No." that is to separate different parameters of an application. Multiple parameters are placed in the numeric order in the ID string. For instance:

xxxx@@cmd0n@@Application 1 title@@cmd**0**v @@Parameter 1 value@@cmd**1**v@@Parameter 2 value@@cmd**2**v@@Parameter 3 value@$c0cmd$ @@cmd1n@@Application 2 title@@cmd**0**v@@Parameter value@$c1cmd$@.....

The "**Parameter value**" is the parameter reference to the corresponding application when the service platform on the terminal device is trying to run this application. The corresponding parameters of the interactive applications supported in the current version of the proposed service generation solution are listed in Table 2.

From the definition of the ID, one of the advantages can be that this type of ID supports multiple applications with multiple parameters. More precisely, such interaction presentation method allows multiple applications running within the lifecycle (press a component of the service – get reactions) of one interaction between the service and the user.

TABLE 2: CORRESPONDING PARAMETERS OF SUPPORTED INTERACTIVE APPLICATIONS

Applications	ID Configuration	
	Application Title	Parameter value
Media Control	Playvideo	URI of the target video (e.g. "medias/clip1.mpg")
	Stopvideo	"null"
Live data feed	Realtimedata	URI of the target data source (e.g. "realtimedata.txt")
Live voting	Voteclient	voting target name__vote_vote value (e.g. "medias/clip1.mpg__vote_good")
	Votereport	Order of request (e.g. "yes")

Therefore by using the XHTML for the visual presentation and "ID Event" method for the interaction presentation, designers can successfully create integrated MDTV service pages with the assistance of the semi-automatic creation tool. Table 3 lists the details of the proposed MDTV service presentation method, which provides references during the generation of all types of service components (The CSS functionality will be incorporated in future work).

TABLE 3: PRESENTATION METHOD BASED ON XHTML AND JAVA ME

MDTV service components			Format and implementation method
MDTV content	Text		XHTML
	Image		Bitmap
	Audio and Video		MIDP MMAPI
	Layout		XHTML+CSS
MDTV interactive applications	Local interaction		MIDP
	Remote interaction	Hyperlink event	XHTML
		ID Event	ID Event method

When adopting the proposed service presentation method in the implementation process (as discussed in Section II A), such method can be a part of the MDTV service application layer agreement. Since the proposed method is compatible with most of the current MDTV standards, the CP and MSP can thus employ it as a universal reference for service generation upon any possible bearer network, even regardless whether the corresponding MDTV standard has its own presentation method. As to the terminal device manufacturer (TDM), no ECMAScript engine is required when developing a service platform/browser that only an interface between XHTML and Java ME is needed in the platform to run interactive applications, as long as the terminal device supports Java ME. The inter-compatibility can thus be brought to most of the MDTV services during their service generation and implementation where content and applications from different MDTV networks can be easily shared without additional service generation or complicated service amendment. Meanwhile, the selected presentation technologies assist in reducing the technical demands during MDTV service generation and the terminal device software production since they are already designers are already familiar with them from having them used before in the IT field.

D. Comparison to other solutions and discussion

When compared to the current commercial solutions and other related research projects, the proposed semi-automated service generation solution exhibits several advantages.

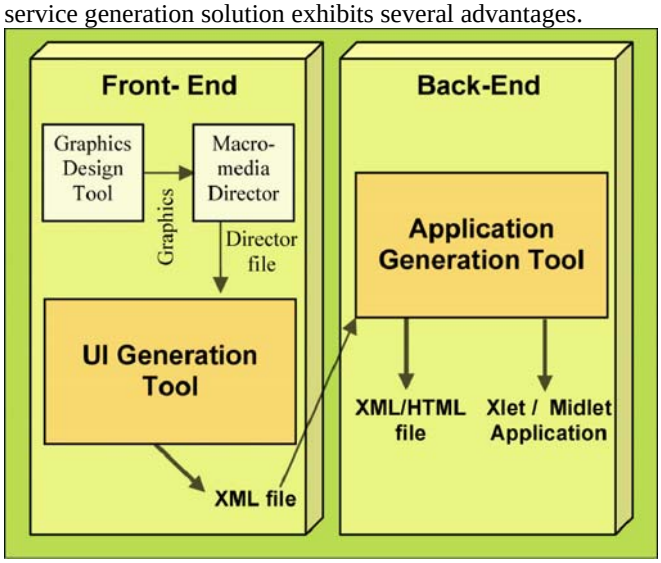

Fig 8: INSTINCT project service generation process ([11] Figure3)

A previous project called INSTICT (IP-based Networks, Services and Terminals for Converging systems) has also proposed a similar DTV service generation process. INSTINCT [12] was a European project in line with the objectives of DVB Convergence of Broadcast and Mobile Services (DVB-CMBS). It was aimed at assisting DVB in realizing the commercial provision of convergent service in mobility with special focus on the DVB-T, DVB-H and DVB-MHP standards in conjunction with the concept of wireless communications networks (notably GPRS and UMTS) combined with terrestrial DVB broadcast networks.

In the INSTINCT project, the service generation process includes two steps (as illustrated in Fig 8): the front-end for the UI components generation and the backend for the interactive components generation. On the front-end generation segment, the designer firstly creates the graphical components by using commercial graphics authoring tool such as Photoshop, in order to prepare original materials for the UI design. With these graphical components, the designer then uses Macromedia Director to design the prototype of the service UI and layout. The project proposed UI Generation Tool is then used by the designer to convert the Director formatted service UI into a custom XML language designed specifically for that project. On the backend generation segment, the designer can utilize the project's proposed Application Generation Tool to further edit the XML formatted service UI by mapping iTV applications (e.g. interactive application) to the UI components according to the service design requirements. Finally, the Application Generation Tool assists the designer in converting the XML formatted iTV service into the requested format (such as HTML format) to finally output the integrated version of the iTV service.

Fig 9: Proposed semi-automated service generation process

Comparing the above process with the proposed process (as illustrated in Fig 9), the Paper's solution has adopted a similar methodology as the INSTINCT solution during the back-end interaction creation. However on the front-end UI creation segment, the proposed semi-automated service generation process shows advantages on several aspects:

The XHTML has been adopted instead of the customize XML language. This is an important advantage when compared with the INSTINCT solution. This is because generally the XML language format is not suitable for visual presentation and XML based visual presentation technologies usually requires extra parsers or rendering engines to layout and present the described by the XML content. Thus in the INSTICT solution, a customized XML schema is introduced for handling the XML UI generation. However, this customized schema is not compliant fully to the current DTV/MDTV service implementations and thus additional tools have been developed to reformat it to be more compatible to the bearer standards. Extra time and effort has to be invested in this case especially for all the conversion processes. In contrast, the proposed process involves widely-adopted XHTML technology that is compatible with most of the DTV/MDTV standard. The proposed solution strictly follows the syntax in W3C XHTML specification during the service generation without creating any additional protocol or schema. Therefore, no extra conversion process is needed and the proposed service creation process is thus shorter and less complicated.

Secondly, by utilizing XHTML, the "Marcomedia Director" step and the "UI Generation Tool" step of the INSTINCT solution can be merged and simplified. This is because by using XHTML authoring tools such as Adobe Dreamweaver and Microsoft Expression Web, as well as several other related ones, the designer can perform the UI prototyping and UI design at the same time. Moreover, the XHTML authoring is relatively easier than the work based on Marcomedia Director due to the popularity and availability of HTML authoring tools in the market.

Moreover in the INSTINCT project, the relative testing performed on the UI generation segment has revealed that the efficiency of the conversion from the Director file to the XML file is not as expected. This is due to the Marcomedia Director having various functions and the UI prototyping is only a small subset of them. Thus the conversion may fail unless specific set of design and authoring rules are followed prior to the import of the file in the UI Generation Tool. In other words, the designer must ensure his/her design follows specific requirements and

guidelines when using Marcomedia Director [11]. As to the proposed service generation process, the XHTML technology is much more familiar to most of the design oriented professionals. More importantly, most of the XHTML authoring tools have their main focus concentrated on visual design and the W3C XHTML syntax is always followed during the service creation. Therefore the designer can employ conventional web service design processes and skills without additional attentions to any extra rules during the MDTV service generation.

Looking at several of the commercial MDTV service generation solutions (such as ACCESS NetFront [21], EXPWAY [20] and Streamezzo [22]), it can be realized that they have a heavily proprietary character throughout their solutions. In contrast, the proposed solution manages to avoid those proprietary issues by utilizing open-standard technologies including XHTML and Java ME. Just like Google Co. opening their Andriod OS SDK to the public to encourage its application development, the proposed solution manages to demystify the MDTV service implementation process and open the service generation to the design and development community. This will encourage more creative individuals, such as more designers, to get involved and encourage the development of new applications and services for MDTV in the future.

In the academic field, some of the related research works also realize the same problems and have proposed several solutions. However, most of them are based on Rich Media technologies [13] [14]. The proposed solution in [13] chooses SVG Tiny for visual presentation and Thinlet XUL in cooperation with Java for the interaction presentation. As discussed in previous sections, Rich Media technologies require advanced design skills and do not support different screen sizes well. Moreover the interaction presentation solution is relatively complicated since the solution tends to use XUL to create the link between SVG Tiny and Java for implementing the corresponding interactions. In comparison to this solution, the proposed solution shows evident advantages on both of these two aspects. XHTML is much more adaptive than SVG Tiny when producing the visual presentation; the ID Event method that has a simple definition is much easier to adopt than a script language like the Thinlet XUL. The proposed solution in [14] tends to provide a convertor for MDTV service creation based on different Rich Media technologies such as MPEG LASeR and MPEG BIFS. This solution is also relatively weak especially with regards to the inter-operability issue of Rich Media.

IV. Initial Testing and Evaluation

A series of software test procedures have been conducted to verify whether the software components of the proposed assistant service creation tool are able to achieve their intended functionality and meet the proposed semi-automated service generation solution specified requirements. Since the software tool is at a prototype stage, the test cases were designed and implemented only for qualitative evaluation of the software functional performance, in order to verify and validate the software on an initial level. More test procedures such as quantitative testing will be introduced according to the needs in future development work.

A. Initial software testing results

TABLE 4: TEST ENVIRONMENT CONFIGURATION

Desktop PC components		Test environment configuration
Hardware	CPU	Intel Pentium Dual-core E5200 2.5GHz
	Memory	2GB
Software	Operating System	Windows XP sp3; Windows 7
	Runtime Environment	Java SE Runtime Environment jre v6
	Development Kit	Java SE Development Kit jdk 1.6.0 update 20
	Integrated Development Environment	Eclipse IDE v3.30
	Assistant software	Windows Internet Explorer 7; Windows Wordpad

Component testing, integration testing and system testing have been performed as part of the testing procedures. Testing methods such as white-box testing and black-box testing were applied. Apart from the testing cases designed for validating the functionality of the software, some additional testing cases were also designed for testing some special components or the architecture of the software, such as the graphical user interface (GUI). All the test procedures were implemented within a Desktop PC environment, of which the configuration details are listed in Table 4. All the results collected from the component testing and system testing have shown that the components of the proposed software are able to work properly with each other or together as an integrated software component with minor errors. The proposed software was further tested in the White-box and Black-box testing against all the functions (including the GUI) that are required and to ensure that the tool can assist designers in manipulating and creating MDTV interactive service pages (based on proposed service presentation method) semi-automatically.

Regarding the errors occurred during the testing process, none of them affects the functionality of the proposed software. The full software testing outcomes are out of the scope of the paper and thus are not presented in detail here.

B. Comparison testing

A comparison testing was also conducted, focusing on evaluating the efficiency of the service creation process between the proposed software and two authoring tools selected from MDTV industry. The aim was to collect quantitative data during a mock-up mobile TV service creation process and to evaluate the proposed solution performance in action by professionals. For this reason a group of service designers were invited from the mobile TV service industry field to participate so as to collect their professional comments

Operations	Average time spent (of 5 persons)		
	Proposed tool	INSTINCT tool	Streamezzo WD
Tool familiarisation	3.6mins	5.3mins	7.8mins
Insert text	0.8mins	1mins	1.2mins
Append image	0.7mins	0.7mins	1.5mins
Set Hyperlink	1.2mins	1.2mins	1.5mins
Append video frame	1.5mins	1.1mins	1.5mins
Output page file	0.05mins	1.8mins	2mins
Amend page	2mins	5mins	3.2mins
Total of service creation	9.85mins	16.1mins	18.7mins

and suggestions on the proposed software system including the authoring tool and the new service presentation method.

The test was conducted in a public research office located in University campus. Five professional service designers (3 from junior level and 2 from senior level) were invited and provided with 5 Desktop PCs to conduct the testing. A prerequisite for the participant selection was experience on authoring XHTML/XML content and familiarity with MDTV background knowledge but no prior experience on using any of the target software employed in the test. This was necessary to ensure their competence to the test and more importantly to gather more objective comparison data on the usability of the software. As the comparison tools, INSTINCT project service creation software system was chosen from the academic field, whilst the Streamezzo Workbench Developer (WD) [42] was chosen from the commercial field. The comparison test between the proposed solution and the INSTINCT solution was aimed at further evaluating their actual efficiency in a practical situation. Streamezzo is a popular commercial mobile TV service solution and its WD is the application authoring tool representing the service creation solution of Streamezzo. The purpose of such comparison is to test whether the proposed "XHTML and Java ME" based service presentation method can provide as good functionality as a Rich Media technology based solution and moreover to test whether the proposed service creation solution can provide improvements over the current MDTV service creation process.

During the test, participants were given a period of time to read the instructions of three software and get familiarized with the UI components and basic functions, which would be used during the test. After that they were asked to perform a set of tasks with the three software tools. The entire process was timed and recorded and all the participants were asked to complete an after-task questionnaire. The tasks and their outcomes are presented below:

1) Task 1: Static service page creation

This task focused on creating a plain mobile TV service page with the same visual and functional elements (including 1 line of text, 1 graphic image, 1 hyperlink and 1 video) by using the three candidate software and then output the page as a file. After that the participants were asked to use the tools to amend the output pages by adding exactly the same components and output the pages again. The time spent was recorded is shown in Table 6.

TABLE 6: AVERAGE TIME SPENT DURING TASK 1

As we can see from Table 6, the proposed tool was reviewed as the easiest one to be familiarized with, with participants spending the least time (an average of 3.6 minutes compared to the 5.3 minutes and 7.8 minutes spent on the other two). This is due to the different complexities of the three types of software. The proposed tool was designed as a single application that is based on the framework of a web page authoring software. All the UI components and basic functions are more familiar to the participants, since they are also found on commercial web page authoring tools. The INSTINCT tool includes two applications that handle the static service page creation, which include the Adobe Director (which exports pages in a custom XML format) and an assistant file converter (which converts from the custom XML to plain HTML format). Participants therefore needed to spend more time on getting familiarized with these. The Streamezzo WD development is based on Eclipse IDE, which was familiar to several of the participants. However, the service development language is based on proprietary syntax of XML , which demands more time in learning and using.

The proposed tool allowed a service page with different components (i.e. text, image, hyperlink, etc) to be created in less time when compared to the other two ones. The service presentation method used by the proposed tool is based on XHTML which is widely employed on the web. Designers could thus perform the service page creation directly through the tool in a familiar way. The INSTINCT project applies XML as the basis of service presentation but the initial page creation relies on the Adobe Director tool. Even though the creation process is more visual, the time spent was more when compared to the proposed tool. WD requires a pure XML-based coding operation during the service page creation, when designers coded the page by following the specified syntax and template. It is considered that the unfamiliarization of the new XML language caused the participants the extra time spent on the task.

The proposed tool saved significant time when outputting the service page. Participants could easily output the service page by merely pressing the "save" button and the created page would be automatically saved as an ".html" file in the default directory. The output process was quick (3 seconds). The INSTINCT solution requires the designers firstly to output a ".dir" file after creating the service page with Adobe Director and then convert this file into standard XML format by using the file converter. These two step cost considerable time (5 minutes) for the page outputting. Streamezzo WD outputs its service page in the form of "client package" in order to run the service page on its client service platform and thus a series of "package generation" operations have to be followed before the designers can have the output page file. This process resulted in more time spent (3.2 minutes) to complete the task.

When using the proposed tool participants spent the least time on amending the service page. In the INSTINCT solution, the designers had to redo the page creation from the beginning. They had to redesign the service page by using Adobe Director and then convert the amended page again with the file

converter. This thus requires the participants to spend much more time than using the other two tools. When using WD, although the participant reloaded the page and edited, it still required additional time to repackage the page into a usable Streamezzo service application. As to the proposed tool, the required operations for amending the service page are the same as amending a XHTML based web page, which participants were already familiar with.

Following the discussion above and along with the total service creation time spent, as shown in Table 6, (9.85 minutes for proposed tool, 16.1 minutes for the INSTINCT tool and 18.7 minutes for Streamezzo ED), it can be concluded that the proposed tool is easier to learn, use and more time efficient in creating a static service page when comparing with the INSTINCT tool and the Streamezzo WD.

2) Task 2: Interactive service page creation

This task focused on the creation of a service page with two interactive applications by using the three different authoring tools. Participants were asked to create a service page that included two icon components (icon 1 and icon 2). A key requirement was that by pressing each icon, the corresponding interactive applications would be triggered. The interactive applications included a voting application and a live data feed application that presented the updated voting result. The time spent on this task is shown in Table 7.

TABLE 7: AVERAGE TIME EXPENSE RECORD OF TASK 2

Operations	Average time expense (of 5 persons)		
	Proposed tool	INSTINCT tool	Streamezzo WD
Append 2 icons	1.4mins	3.1mins	2mins
Add voting application to icon 1	1mins	1.3mins	6.8mins
Add live data application to icon 2	1.1mins	1.1mins	7.3mins

As we can see from Table 7, the INSTINCT tool required more time (3.1 minutes) than the other two (1.4 minutes for proposed tool and 2 minutes for WD) whilst the proposed tool required the least. The main reason is considered to be that during the INSTINCT creation solution, the service page has to be converted into an XML file format before any interactive features could be added on the page. Thus as discussed before extra time is required on the page file conversion from the Director file format to XML. One the other hand, it was easier for most of the participants to code in standard XHTML than coding in customized XML with a specific syntax, which reflects further the less amount of time spent when using the proposed tool than Streamezzo WD.

In the remaining steps when the participants were asked to add the two interactive applications to the corresponding two icons, the proposed tool and the INSTINCT tool took the advantages of their special function design, which resulted in considerable time saving. This is because when using these two tools, designers only have to drag the application from the library window, drop it to the target icon and fill in the necessary parameters in order to create the interactive application. At the same time, a special ID string is added automatically to the corresponding icon element in the source code. When creating interactive applications with the proposed tool and the INSTINCT tool, the participants did not have to use any scripting/programming language and thus did not require any programming knowledge to employ it but simply to know how to drag-and-drop a UI element. Whilst with the Streamezzo solution, a proprietary scripting language named Instantscript was needed for implementing the interactive functions. Therefore additional time was required by all participants in order to learn to use such new language before the actual coding.

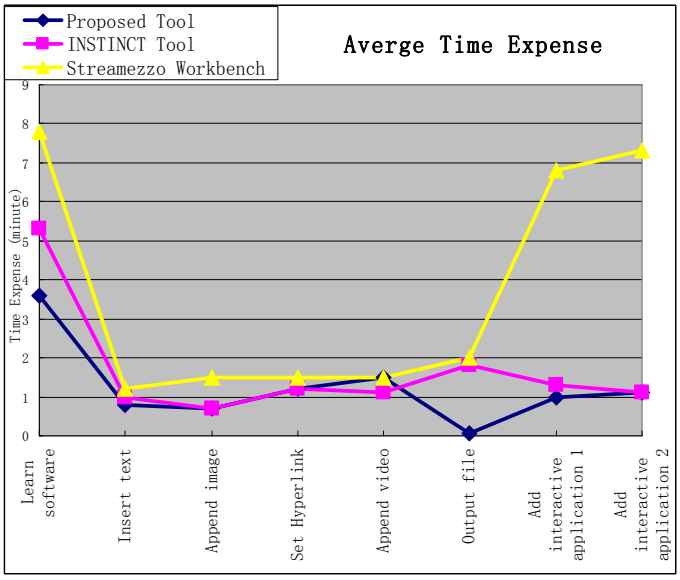

Fig 11: Line chart of average time spent on Tasks 1 and 2

Figure 11 illustrates the average time spent on the service creation operations during Task 1 and Task 2.

3) Task 3: Creating a batch of service pages

Participants were asked to use the three tools to create three service packages with all the necessary functions for a MDTV service including basic service GUI, video playpack and interactive applications such as video control, voting and live data feed. Each package included several service pages with mixed types of service components (text, image, hyperlink, video and interactive application) and participants had to follow the same service requirements when using the different tools to create each package. This test case was designed to simulate more closely to real-life what designers experience at work, and to evaluate the performance ~~eddicieney~~ efficiency of the three tools. The corresponding time spent is shown in Table 8. Figure 12, 13 and 14 illustrate the differences in the time spent between the three software tools during the creation of Service Package 1, 2 and 3.

As shown in Figures 12, 13 and 14, the proposed solution, is always below the other two tools. This shows that the proposed MDTV service creation solution offers relatively high

performance efficiency in terms of service production time when compared to the other two solutions.

TABLE 8: TIME SPENT ON TASK 3

Participant (P)	Time expense on each package (minutes)								
	Proposed tool			INSTINCT tool			Streamezzo WD		
	Pack1	Pack2	Pack3	Pack1	Pack2	Pack3	Pack1	Pack2	Pack3
P1	5.2	4.5	7.3	6.8	5.5	9	8.2	5.2	11.3
P2	4	3.8	8.2	6	5.3	8.5	8.7	5.9	12.1
P3	6.2	3.8	8.5	8.2	7	8.4	7.2	4.5	9.3
P4	5	5.1	6.9	7.5	7.2	9.2	9	8.2	15.5
P5	4.8	4.2	7.9	5.5	4.7	7.8	10.3	9	14

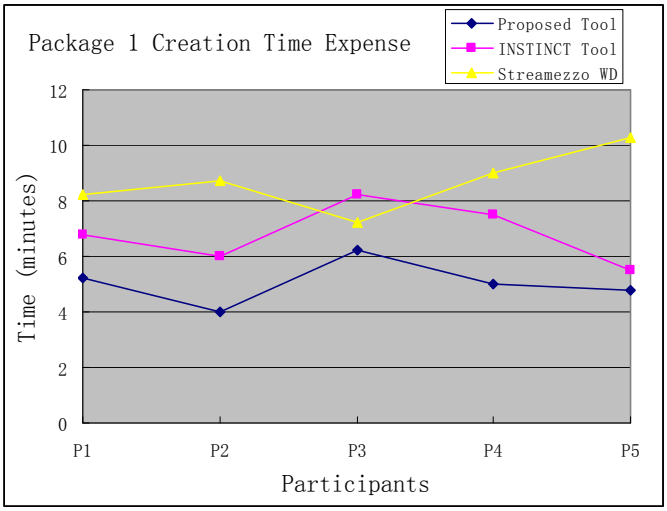

Fig 12: Line chart of time spent on Package 1

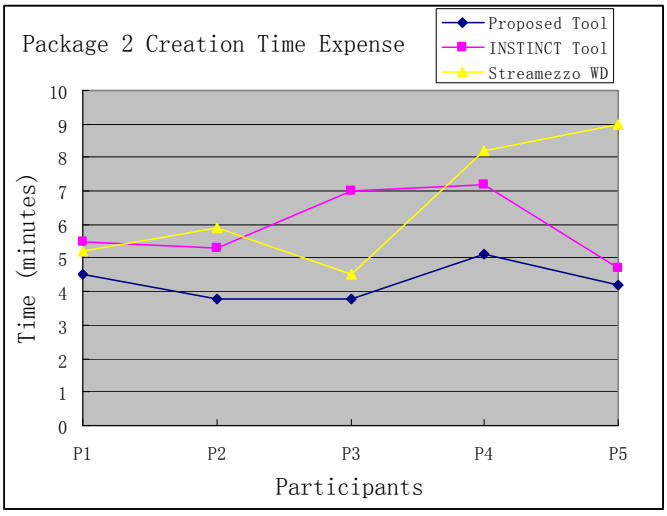

Fig 13: Line chart of time spent on Package 2

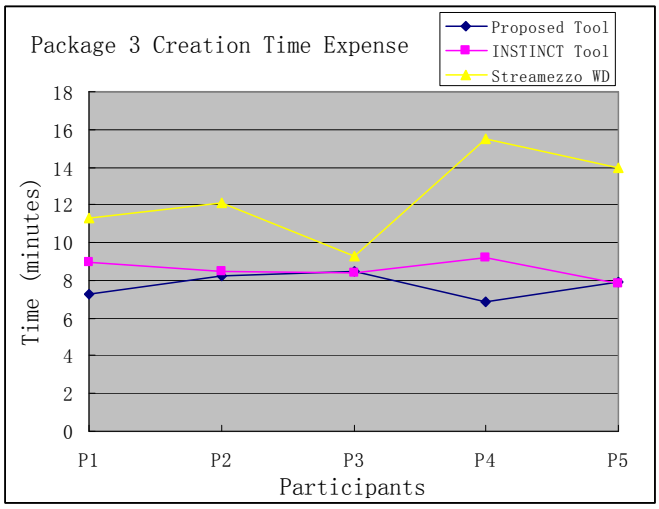

Fig 14: Line chart of time pent on Package 3

4) Questionnaire and feedback

An after-task questionnaire was employed to collect more testing results and feedback. More precisely participants commented positively on the proposed tool's ease-of-use and learn-ability. They also responded that the GUI is clearly laid out and the functions, such as the XHTML coding, interactive application editing and output are simple to handle. They admitted that the tool did facilitate a faster MDTV service production, in that besides the XHTML based coding, only drag-and-drop operations were needed during the service creation. This was thought to be much more convenient than coding with a complicated scripting language such as in the case of the Streamezzo Workbench Developer. Furthermore, the participants highlighted that it is easier to complete all the creation operations within in one piece of software such as in the case of the proposed tool and Streamezzo WD, than using a software suite like INSTINCT solution (Adobe Director and File Converter).

Participants also commented on potential improvements for proposed software tool. They particularly commented that although the software tool's GUI is simple and effective there is a help function and help reference missing which would be a beneficial component. Participants had difficulty in finding any guidelines and troubleshooting guides when they met problems during the tasks. They think it would be more helpful to develop a help system similar to the one Streamezzo has.

In addition to this, participants commented that the software's look-and-feel resembled more a prototype tool than a commercial one. Some bugs were found during the tests and many useful functions such as "save project" that are not currently included they thought that they would be very useful to have. Also, all participants responded that the available interactive applications within the proposed tool are basic and limited in number when compared to the Streamezzo Workbench Developer. More applications such as touch screen based applications should be developed to meet the current market requirements. Additional usability testing data of the proposed tool were collected but these are not reported here as

they are, on one hand, out of the scope of the paper and on the other due to page limitation.

As a result, the proposed semi-automatic service creation tool has been tested in a simulated MDTV service generation process. The XHTML and Java ME based service presentation method has also been tested through comparison testing. The tool was found by participants to be easy to learn and it has been found useful efficient in helping participants manipulating and creating MDTV interactive service pages, regardless of their knowledge and skills in MDTV technologies and software engineering. This means that as long as a designer has some basic knowledge on using computer programs and has basic experience on XHTML authoring, (s)he is able to participate in the MDTV service generation through the proposed tool. The technical demands on the designer are therefore reduced, more design oriented professionals can become involved more easily and the development of the MDTV service can be further encouraged. The proposed presentation method has received positive comments by professional designers and it has been also found particularly cost effective in terms of MDTV service production times.

V. Future Work and Conclusion

Future work will be carried out in order to integrate the proposed solution with mobile TV Electronic Service Guide (ESG) function. A new function would be added to the proposed creation tool so that the service page metadata file can be also generated and outputted automatically during the service generation process. Furthermore and with regards to the proposed service presentation method, SVG will be incorporated as an assistant graphics technology to the Bitmap format since SVG can be more efficient from a storage and transmission bandwidth perspective.

This paper has argued that there is a need for the production and availability of high quality of multimedia and interactive applications in the current MDTV service market. However the current service generation solutions do not satisfy this fully. This paper has proposed a novel multimedia and interactive service generation solution to cope with service production issues in the current MDTV industry field.

The proposed semi-automatic MDTV service creation process introduces a universal MDTV service generation solution. The semi-automatic service creation tool and the novel MDTV service presentation method have been the main components of the proposed solution. The solution is compatible with most of the leading MDTV standards in the market.

The proposed XHTML and Java ME based service presentation method takes advantage of these two mature technologies by providing excellent compatibility, scalability and functionality. It encourages even more design-oriented professionals participating into the MDTV service development. The development this method on one hand improves the inter-compatibility of outputted service content through various MDTV standards and on the other hand encourages an open-standard approach that opposes the proprietary character of current MDTV service implementation solutions.

The proposed semi-automatic service creation tool provides a convenient development environment to the designers. Its compact GUI and proper functions design make it easy to learn and use. It simplifies the operations during the service creation by replacing most of the coding works with drag-and-drop gestures. Less technical demand is placed on the designers, allowing them to participate more in the service production process and thus create higher quality services.

Testing revealed that the proposed semi-automatic service creation solution including the software tool and the "XHTML and Java ME" based presentation method is easier to learn than the INSTINCT and Steramezzo corresponding software tools. In addition to this, the proposed software tool enabled participants to achieve a faster MDTV service creation than the other two candidates. This is more remarkable when creating the interactive applications, especially as no scripting knowledge and coding is required.

References

[1] Moxian Liu, "Semi-automated Mobile Television Interactive Application Generation based on XHTML and Java ME", PhD dissertation, School of Engineering and Design, Brunel University, London, UK, 2011.

[2] Moxian Liu, E. Teskleves, J.P.Cosmas, "Semi-automatic creation of graphically-rich mobile Television services and applications using an XHTML browser and J2ME", in *2010 IEEE International Symposium on Broadband Multimedia System and Broadcasting (BMSB)*, Shanghai, 2010, pp.1-7.

[3] Artur Lugmayr et al, *Digital Interactive TV and Metadata: Future Broadcast Multimedia*, New York: Springer-Verlay, 2004,

[4] Dr Ulrich Reimers, *DVB: The Family of International Standards for Digital Video*, Broadcasting, Second Edition, Germany: Springer, 2005.

[5] H. Song, J Park, "Design of an Interoperable Middleware Architecture for Digital Data Broadcasting", *IEEE Transactions on Consumer Electronic*, Volume 52, pp1433-1441, 2006.

[6] M.M.Saleemi, et al, "System Architecture and Interactivity Model for Mobile TV Applications", in *3rd International Conference on Digital Interactive Media in Entertainment and Arts*, 2008.

[7] R. Schatz, S. Wagner, A. Berger, "AMUSE – A Platform for Prototyping Live Mobile TV Services", in *Mobile and Wireless Communication Summit 2007* 16th IST, pp1-5, July 2007.

[8] P. Leroux, et al, "UIML Based Design of Multimodel Interactive Applications with Strict Synchronization Requirements", in *2009 Second International Conferences on Advances in Computer-Human Interactions*, pp175-180, 2009.

[9] E Marilly, G Delegue, O Martiont, "Rich Media Service Creation for Interactive Mobile TV", *The 2007 International Conference on Next Generation Mobile Application, Services and Technologies* 12-14 Sept. 2007 Page(s):85 – 90.

[10] *ECMAScript Mobile Profile: A Wireless Markup Scripting Language Approved version1.0 – 20 Oct 2006*, Open Mobile Alliance Ltd standard, 2006.

[11] E. Tsekleves, et al, "Semi-automated Creation of Converged iTV Services: From Macromedia Director Simulations to Services Ready for Broadcast", *Journal of Virtual Reality and Broadcasting*, Volume 4, no.17, 2007.

[12] INSTINCT. (2004). IP-based Networks, Services and Terminals for Convergence Systems. [Online]. Available: http://www.ist-instinct.org/.

[13] M. Spika, "An XML-based Software Platform for DVB-H and IP Datacast with Full Java-logic Capability", in *International Symposium on Consumer Electronic (ISCE 2008)*, pp1-4, April 2008.

[14] Emmanuel Marilly, et al, "Rich Media Service Creation for Interactive Mobile TV", in *2007 IEEE International Conference on Next Generation Mobile Application, Service and Technologies (NGMAST 2007)*, pp85-90, 2007.

[15] F. Allamandri, et al, "Service Platform for Converged Interactive Broadband Broadcast and Cellular Wireless", *IEEE Transactions on Broadcasting*, Vol.53, No.1, March 2007.
[16] B. Lee, J. Song, Y. Nam, "Converged Mobile TV Services Supporting Rich Media in Cellular and DVB-H Systems", *IEEE Transactions on Consumer Electronic*, Vol. 54, Issue 3, Page(s) 1091-1097, August2008.
[17] Information technology – Coding of audio-visual objects – Part 20: Lightweight Application Scene Representation (LASeR) and Simple Aggregation Format (SAF), ISO/IEC JTC 1/SC 29, ISO/IEC FDIS 14496-20:2006(E), 2006-01-23.
[18] *MPEG-4 BIFS White Paper*, International Organization for Standardization ISO/IEC JTC 1/SC 29/WG v11, Coding of Moving Pictures and Audio, October 2005, [online] Available: http://mpeg.chiariglione.org/technologies/mpeg-4/mp04-bifs/index.htm
[19] *ECMAScript Language Specification, ECMA-262*, 5th Edition/December 2009.
[20] EXPWAY, "Mobile TV products" [online]. Available: http://www.expway.com/mobile-tv.php.
[21] ACCESS, "NetFront Browser DTV Profile DVB-H Edition: Deliver an Innovative and Consistent Mobile DTV Experience Across Multiple Mobile Platform". Available: http://www.access-company.com/products/mobile_solutions/mobile_tv/dvb-h_profile.html.
[22] Streamezzo, "Streamezzo Interactive Mobile TV". Available: http://www.streamezzo.com/eng/solutions/mobile-tv.php.
[23] *Part1 – ATSC Mobile Digital Television System, Part 4 – Announcement, Part 5 – Application Framework, Part 6 – Service Protection, Advanced Television Systems Committee*, ATSC-Mobile DTV Standard A/153, 2009.10.
[24] *ARIB STD-B23 v1.1 English Translation (2004)*, Application Execution Engine Platform for Digital Broadcasting, Association of Radio Industries and Businesses, available online: http://www.arib.or.jp/english/index.html/.
[25] *Digital Audio Broadcasting (DAB); Middleware; Part1: System aspects*, European Broadcasting Union ETSI standard ETSI TS 102 365 – 1 v1.1.1, 8/2009.
[26] *Digital Audio Broadcasting (DAB); Middleware; Part 2: DAB*, European Broadcasting Union ETSI standard ETSI TS 102 365 – 2 v1.1.1, 8/2009
[27] *Universal Mobile Telecommunications System (UMTS); LTE; Dynamic and Interactive Multimedia Scenes (DIMS) (3GPP TS 26.142 version 9.0.0 Release 9)*, European Broadcasting Union ETSI standard, ETSI TS 126 142 v9.0.0, 2009
[28] *Rich Media Environment Technical Specification Candidate Version 1.0 – 14 Oct 2008, OMA-TS-RME-V1_0-20081014-C*, Open Mobile Alliance standard, 2008.
[29] *White Paper on Rich Media Environment Technology Landscape Report Candidate – 14 Oct 2008, OMA-WP-Rich_Media_Environment-20081014-C*, Open Mobile Alliance standard, 2008.
[30] Nokia. (2007). Nokia DVB-H Mobile TV Implementation Guidelines, Release 3.0, Air Interface. [Online]. Available: http://www.mobiletv.nokia.com/solutions/airif/mtvig_documents_3-0.php.
[31] Silicon & Software Systems. onHandTV Production Overview. [Online]. Available: http://www.s3group.com/tv-technology/tv-products/onhandtv/.
[32] Thin Multimedia, Inc (tmi). thinMobileTV Middleware. [Online]. Available: http://www.thinmultimedia.com/products/thinmobile_tv/.
[33] TATA ELXSI Engineering Creativity. Tata Elxsi Product Solution, DVB-CMBS stack. [Online]. Available: http://www.tataelxsi.com/htmls/pds/pds_mobile_TV.htm.
[34] W3C Recommendation. (2008, December 22). Scalable Vector Graphics (SVG) Tiny 1.2 Specification. [Online]. Available: http://www.w3.org/TR/SVGTiny12/.
[35] Adobe. Flash Lite. [Online]. Available: http://www.adobe.com/products/flashlite/.
[36] W3C. XML Essentials. [Online]. Available: http://www.w3.org/standards/xml/core.
[37] W3C Recommendation. (2008, July 29). XHTML Basic 1.1. [Online]. Available: http://www.w3.org/TR/2008/REC-xhtml-basic-20080729/.
[38] *Digital Video Broadcasting (DVB); Globally Executable MHP (GEM) Specification 1.2.2 (including IPTV)*, DVB Project Document A139 r4, June 2009.
[39] *White Paper on Rich Media Environment Technology Landscape Report Candidate – 14 Oct 2008, OMA-WP-Rich_Media_Environment-20081014-C*, Open Mobile Alliance standard, 2008.
[40] HbbTV, "Hybrid Broadcast Broadband TV" [Online], Available: http://www.hbbtv.org/
[41] Java - Sun Microsystems. Mobile Information Device Profile (MIDP); JSR137, JSR118 Overview. [Online]. Available: http://java.sun.com/products/midp/overview.html.
[42] Streamezzo, "Streamezzo Workbench Developer" [Online], Available: http://www.streamezzo.com/eng/products/stz-workbench.php
[43] Glenford J. Myers, *The ART of SOFTWARE TESTING, Second Edition*, John Wiley & Sons, Inc, 2004.
[44] Shari L. Pfleeger, *Joanne M. Atlee, Software Engineering: Theory and Practice, Third Edition*, Pearson Education, 2006.
[45] B. Lee, J. Song, Y. Nam, "Converged Mobile TV Services Supporting Rich Media in Cellular and DVB-H Systems", *IEEE Transactions on Consumer Electronic*, Vol. 54, Issue 3, pp1091-1097, August2008

Dr. Moxian Liu received his BEng in Electrical Engineering and its Automation in Sichuan University China in 2004. He was awarded his M.Sc. degree in Data Communication Systems at Brunel University UK in 2006 and earned his PhD degree on Digital Multimedia Broadcasting in Brunel University in 2011. His research interests are on the middleware and software layers of DTV and MDTV.

Dr. Emmanuel Tsekleves is a Lecturer in Multimedia Design and Technology at Brunel University. His PhD Thesis was on the semi-automated creation tools for the production of iTV services. His research interests lies in the area of multimedia service creation and user-generated content technologies for broadcast and broadband networks, mobile Television, 3D Tools and Applications.

John Cosmas is a Professor of Multimedia Systems at Brunel University. He received his PhD in Image Processing at Imperial College in the UK. His research interests are focused on content and service creation and management systems for converged broadcast and telecommunication systems and the Future Internet.