2IMMERSE Production Suite: A Platform for Creating Interactive Multi-Screen Experiences

Thomas Röggla[1], **Jie Li**[1], **Jack Jansen**[1], **Andrew Gower**[2], **Martin Trimby**[2], **Pablo Cesar**[1,3]
[1]Centrum Wiskunde & Informatica (NL), [2]BT Technology (UK), [3]Delft University of Technology (NL)
{t.roggla, jie.li, j.jansen, p.s.cesar}@cwi.nl, {andrew.p.gower, martin.trimby}@bt.com

ABSTRACT

We present a software solution for creating and playing back interactive multi-screen experiences. The system consists of a pre-production application for editing layout and timing of interactive media objects and a live-triggering software for inserting on-demand content during live streams of these edited experiences. The system is governed by a hierarchical file format that defines the temporal relationship and synchronisation of media objects. We also briefly introduce the concept of DMApp Components, an open specification which is used to describe and create custom interactive media objects.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Author Keywords

Multi-screen experiences; object-based broadcasting; multimedia authoring; immersive media

INTRODUCTION

In the world of traditional broadcasting, television content is assembled at a TV studio, or in case of a live event, on-site and is then broadcast to people's homes via cable or through airwaves. This content is static and solely meant for passive consumption on a television screen. Object-based broadcasting extends this concept by delivering all media objects to be displayed separately, for instance via the Internet, and assembling them dynamically at the end-user [1], as illustrated in Figure 1. Through this, the viewer has more choice and control over the content shown on the screen and it affords them more interactivity. With the work presented in this paper, we want to take this approach one step further. We present an end-to-end software suite [4] which allows content producers to more easily create object-based broadcasting content, spanning over multiple screens, with which the viewer can interact and get immersed in more meaningful ways [5]. For example, while a motorcycle race is being played on a communal television screen, viewers can obtain information about riders on their mobile devices or chat with other people watching the same programme during the intermission [3]. Moreover, apart from the pre-production of content, the system also has support for inserting on-demand snippets of content into live broadcasts, i.e. a crash during a motorcycle race, by means of a live-triggering tool.

The Adjunct Proceedings of ACM TVX 2018, Seoul, Republic of Korea, June 2018. Copyright is held by the author/owner(s).

At the very core of this is a hierarchical XML file format which dictates the timing of media objects/interactive content [2]. The format supports sequential and parallel composition and synchronisation of objects on a timeline, while the laying out of objects on the screens is controlled by a layout engine. This layout engine is able to responsively arrange objects on screens of various sizes and orientations. So for example, on a big screen, it may be easily possible to nicely render a motorcycle race's grid positions, whereas on a small screen this would take up too much space. The layout engine is able to account for these cases intelligently.

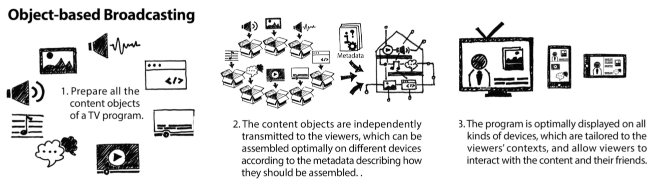
Object-based Broadcasting
1. Prepare all the content objects of a TV program.
Metadata
2. The content objects are independently transmitted to the viewers, which can be assembled optimally on different devices according to the metadata describing how they should be assembled. .
3. The program is optimally displayed on all kinds of devices, which are tailored to the viewers' contexts, and allow viewers to interact with the content and their friends.

Figure 1. The process of object-based broadcasting

In this demo, we will be showing an interactive example broadcast consisting of a recording of the 2017 MotoGP race in Silverstone in England on a large television screen and one or more mobile devices. In the interest of time the experience that will be shown has been pre-edited using our pre-production software. The main focus of the demo will lie on the live-triggering tool, which will be used to insert on-demand snippets and interactive components into the running stream.

SYSTEM ARCHITECTURE

The system described in this document is made up of several components working together in a microservice architecture. While the system is comprised of roughly 20 independent components working together, three of them are responsible for the bulk of the work, namely: timeline, layout and authoring. The timeline service is responsible for executing a timeline document supplied by the authoring application. This timeline document is a hierarchical, XML-based format for governing temporal relationships between playback items. The format contains references to objects, in this context termed *DMApp Components*, which at their most basic level, are self-contained pieces of HTML, CSS and JavaScript and can satisfy any arbitrary use-case, as long as it is supported by a web browser.

Whereas the aforementioned services run without user interaction and are responsible for experience playback, the authoring

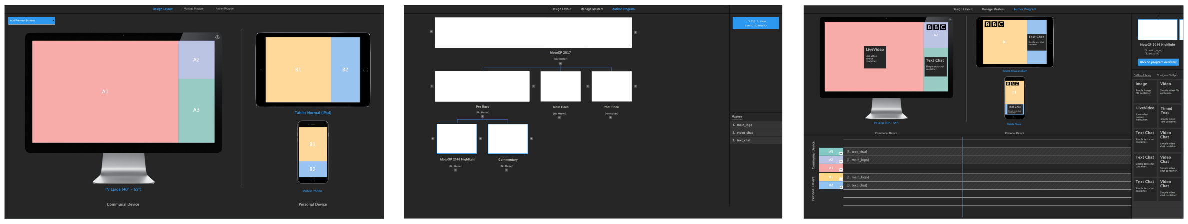
Figure 2. Pre-production workflow (from left to right): layout design, chapter creation, timeline editing

application is the part of the system that a production team would use to create these experiences. The authoring application is a browser-based application, with which a producer can assemble an experience in a step-by-step fashion. Roughly speaking, this process is comprised of *designing the layout*, *creating masters*, *defining chapters* and *designing the timeline* (see Figure 2). In the layout design phase, the user can either load a predefined layout, e.g. from a previously created experience, or add screens manually and segment them into regions using the mouse. The master creation is concerned with the creation of so-called *masters*. This is akin to the concept of master-slides in a presentation program like PowerPoint. So for instance, if the user wants certain objects to be shown all throughout the experience, like for instance the broadcaster logo and the lap-counter during a motorcycle race, they would create a master layout containing DMApp Components rendering these objects. This avoids the need to redefine these components for every chapter of the presentation. The utility of this approach will become apparent in the next paragraph.

In the next step of the process, the producer can define a tree-based hierarchy of so-called chapters. To borrow from the example of a motorcycle race again, such an experience might have one root chapter and three child-chapters: pre-race, main event and post-race discussion. The creator of the experience might then assign a master layout defining a broadcaster logo to be shown in the corner of the screen to the root chapter. This has the effect that these components cascade down to the child-chapters and do not need to be redefined.

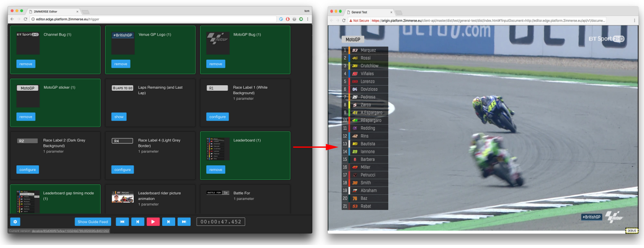
Figure 3. The live-triggering tool in action

Finally, leaf chapters in this hierarchy can be opened in the builtin timeline editor. This is very similar in concept to industry-standard video editing applications, where one arranges media items items on multiple timeline tracks. The key difference here is that there is a timeline track for every region in every screen defined in the previous steps. Moreover, instead of media items, the user can drag and drop DMApp Components onto timeline tracks and arrange them. In addition to that, the timeline editor can also be used to create snippets of timeline in a similar fashion. These snippets can then be triggered on-demand during broadcast using the live-triggering tool, e.g. a rider crashes during a motorcycle race.

EXPERIENCE PLAYBACK

After the editing process, the experience is ready to be played back on the infrastructure. This is done by navigating to a web address on the television and by installing the experience app on mobile devices. The broadcast then plays like a traditional programme, with the key difference that the viewer receives additional interactive content on their companion device and has the possibility to interact with other people watching the experience.

On the production side, the content producer now has the ability to insert *events*, snippets of timeline defined during pre-production process, into the ongoing broadcast using the live-triggering tool (Figure 3). These events can be as simple as showing a box with text on the screen or be more complex sequences such as playing back a replay together with entry- and exit-animations. These on-demand events are governed by the same rules as the rest of the timeline document.

Just like the pre-production application, this live-triggering tool is web-based. It lists all the events that have been defined for the experience along with possible parameters: For instance, a URL to a replay clip or the name of a rider in a race. After all required parameters have been filled in, events can be launched with the click of a button. The events are then inserted into the stream immediately and played back. Active events are highlighted in green.

DISCUSSION

In this paper, we have described and end-to-end solution for creating interactive multi-screen experiences. The system infrastructure is comprised of a series of services managing interactive broadcasts whose timing and synchronisation is managed using a hierarchical file format. Furthermore, we presented a web-based authoring application for creating said experiences, which are then played back by the infrastructure. Part of this authoring suite is a live-triggering tool, which allows the content producer to insert predefined content into a running broadcast.

ACKNOWLEDGEMENTS

The work presented in this paper was supported by the EU funded H2020 ICT project 2-IMMERSE, under contract 687655

REFERENCES

1. Mike Armstrong, Matthew Brooks, Anthony Churnside, MEF Melchior, and M Shotton. 2014. Object-based broadcasting-curation, responsiveness and user experience. (2014).

2. Jack Jansen, Pablo Cesar, and Dick Bulterman. 2018. Workflow Support for Live Object-Based Broadcasting. In *Proceedings of the ACM symposium on Document Engineering (DocEng '18)*. ACM, New York, NY, USA.

3. Ian Kegel, James Walker, Mark Lomas, Jack Jansen, and John Wyver. 2017. 2IMMERSE: A Platform for Orchestrated Multi-Screen Entertainment. In *Adjunct Publication of the 2017 ACM International Conference on Interactive Experiences for TV and Online Video (TVX '17 Adjunct)*. ACM, New York, NY, USA, 71–72. DOI: http://dx.doi.org/10.1145/3084289.3089909

4. Jie Li, Thomas Röggla, Jack Jansen, Maxine Glancy, and Pablo Cesar. 2018a. A New Production Platform for Authoring Object-based Multiscreen TV Viewing Experiences. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA.

5. Jie Li, Zhiyuan Zheng, Britta Meixner, Thomas Röggla, Maxine Glancy, and Pablo Cesar. 2018b. Design an Object-based Preproduction Tool for Multiscreen TV Viewing. In *Proceedings of the 2018 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '18)*. ACM, New York, NY, USA.