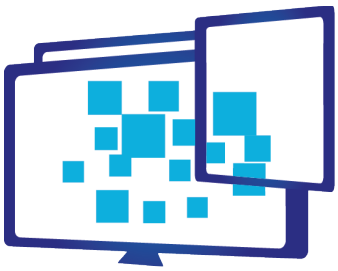

MediaScape

Dynamic Media Service Creation, Adaptation and Publishing on Every Device

Instrument: Collaborative Project (STREP)
Call Identifier: FP7-ICT2013-10
Grant Agreement: 610404

WP7 – EXPLOIT – Dissemination, Standardisation and Exploitation

Deliverable Number	D7.4
Title	Final Standardisation Report
Version	1.0
Date	29 April 2016
Status	FINAL

EC Distribution

Project Partners: VIC, IRT, NEC, BBC, W3C, NOR, BR

Contributors: **W3C, VIC, NOR, BBC, IRT**

Document Control

Version	Date	Author	Modifications
0.1	22 October 2015	François Daoust (W3C)	First outline based on D7.3
0.2	9 March 2016	Chris Needham (BBC)	Status update of BBC standardisation activities
0.3	14 March 2016	François Daoust (W3C)	Updates on main contributions to standardisation activities started at W3C
0.4	16 March 2016	Ingar M. Arntzen (NOR)	Cross-device activities in the Multi-Device Timing Community Group
0.5	17 March 2016	Alexander Erk (IRT)	RadioWEB updates
0.6	28 April 2016	Mikel Zorrilla (VIC)	Status update of VIC standardisation activities
0.7	29 April 2016	Alexander Erk (IRT)	Document review
0.8	29 April 2016	François Daoust (W3C)	Finalized updated action plan
1.0	29 April 2016	Mikel Zorrilla (VIC)	Minor changes

Table of Contents

1. Introduction

This document reports on the standardisation activities of the MediaScape project throughout the life of the project. It completes and replaces deliverable D7.3 Initial Standardisation Report, which reported on standardisation activities for the first 17 months of the project.

1.1. Background

HTML5 [HTML5] has turned the Web into a global application platform, available on each and every device. An ever increasing range of devices are capable of running HTML5 applications, from traditional computers, tablets and mobile phones to smart TVs, radio receivers, smart watches and other consumer electronic devices. Experience shows that end users use some of these devices in combination in so-called second-screen scenarios, e.g. browsing the Web on their tablet while they watch TV. However, when the MediaScape project started, there was no easy and standardized way for a Web application to leverage the availability of multiple devices and create a consistent and seamless user experience across devices.

1.2. Objectives

MediaScape seeks to facilitate the marriage of the TV, PC and Mobile worlds through a standard Web-based solution that includes real-time delivery and synchronisation of media contents and applications/services across a variety of devices.

The main objective of MediaScape's standardisation activities are to contribute to the standardisation of technologies required and developed by the project, within relevant standardisation organisations. These technologies are the outcomes of the technical work packages:

- WP3 – Multi-connection and authentication through multiple devices
- WP4 – Shared Application runtime and Synchronisation
- WP5 – Multi-device Adaptation

Project partners developed prototypes in WP6 (see D6.1 Initial Prototypes and D6.3 Final Prototypes) to gain experience of these technologies. Prototyping helps filter technologies that are good candidates for standardisation from those that need further research. As such, standardisation activities built on the results of the prototyping task.

By definition, successful standardisation requires engagement from main relevant actors (both inside and outside MediaScape). The task thus included the organisation of a standardisation workshop on “Multi-Device Web Applications” to gather support from a wide range of companies. Dissemination activities (see D7.1 Dissemination Strategy and D7.2 Final Dissemination Activities) and communication with other relevant EU projects (managed in WP1) contribute to that effort as well.

1.3. Structure of this document

Section 2 outlines the standardisation landscape that is directly relevant to MediaScape. The section highlights technical gaps that MediaScape identified in existing standards, and the initial list of contributions that project partners considered as a result.

Section 3 details the actual contributions from MediaScape towards standardisation activities, including the organisation of a standardisation workshop, the publication of the Cross Platform Authentication specification by the EBU, and the creation of two groups in W3C. Standardisation activities followed the initial action plan, described in D6.3 Initial Standardisation Plan.

Standardisation takes time and most of the standardisation efforts will continue beyond the end of the project. Section 4 discusses the standardisation action plan at the end of the project, including contributions that MediaScape partners anticipate after the end of the project.

2. Standardisation plan

As stated in the introduction, MediaScape seeks to facilitate the marriage of the TV, PC and Mobile worlds through a standard Web-based solution that includes real-time delivery and synchronisation of media contents and applications/services across a variety of devices. Technologies required or developed as part of this Web-based solution form the initial list of candidate topics for standardisation. After a quick overview of relevant standardisation organisations for MediaScape, this section reviews the list of technologies that are or were initially thought to be directly relevant to MediaScape, pointing out related standardisation efforts and potential gaps that were identified at the beginning of the project. Progress and changes of perspective during the project are mentioned.

The technical gaps led to the creation of an initial action plan, defined in D7.3 Initial Standardisation Report. That plan was slightly updated based on the final outcomes of the different work packages. This section reports on the adjustments made to the plan.

Note that actual contributions to standardisation activities are detailed in section 3.

2.1. Relevant standardisation organisations

MediaScape develops a solution based on Web technologies. As such, although the project will use a number of Internet protocols behind the scenes, only SDOs that operate at the application/service layer are listed as directly relevant to the project, while SDOs that operate at lower layers of the stack such as IETF or DVB are not mentioned.

2.1.1. W3C

The W3C develops royalty free Web standards such as HTML5, CSS, and various JavaScript APIs that together form the Web platform, implemented in Web browsers on a wide range of devices (mobiles, tablets, desktops, TV, and more connected devices). W3C standards are also at the core of iTV middlewares such as HbbTV or HybridCast.

There are 3 main types of groups in W3C:

- **Working Groups (WG)** are the only group doing standards-track work. They require W3C membership and commitment to W3C's Royalty Free patent policy.
- **Interest Groups (IG)** focus on identification of use cases, requirements and gap analysis to then in turn provide to relevant Working Groups. They also require W3C membership but no strong IPR commitments since Interest Groups do not produce technical specifications.
- **Community Groups (CG)** are open to anyone, and anyone can create a Community Group. Community Groups focus on pre-standards opportunities within new communities of interest, sometimes drafting technical specifications that can later on transition to a Working Group. In other words, a Community Group is a forum for discussion on some technical topic with a clear royalty free IPR regime.

Not surprisingly, most technologies that emerge from MediaScape are good candidates for standardisation efforts within W3C, and W3C is involved in the project precisely for that purpose. More specifically, when the project is started, the following groups have been identified as particularly relevant to MediaScape.

2.1.1.1. Device APIs Working Group

The Device APIs Working Group[1] is chartered to create client-side APIs that enable the development of Web Applications that interact with device hardware, services and applications such as the camera, microphone, system sensors, native address books, calendars and native messaging applications. The Network Service Discovery API [NSD] seemed of particular relevance for MediaScape, as it enabled the discovery of devices and services on the local network directly from within the Web browser.

Rich Tibbett, the editor of the Network Service Discovery API from Opera Software, has worked on a different proposal called Network Web Sockets[2]. This proposal makes it possible for an application running in a

[1]Device APIs Working Group home page: http://www.w3.org/2009/dap/

[2]Network Web Sockets proposal: https://github.com/namedwebsockets/networkwebsockets

Web browser to discover and exchange messages with another application running in another Web browser provided they agree on the name of the socket used to exchange messages. The proposal puts the emphasis on security: a third party service listening on the network will not be able to connect with any of the applications unless it knows the name of the socket.

MediaScape partners developed discovery prototypes based on these two proposals (IRT for the Network Service Discovery, BBC for Network Web Sockets). For Network Web Sockets, given the need to agree on a socket name, the prototype assumes that applications that need to discover each other will both listen on a well-known "mediascape" socket, thus building discovery on top of Network Web Sockets.

However, the Network Service Discovery specification did not gain traction among implementers in the end. Giving Web applications the power to 1) list and 2) interact with devices found on the local network raises security issues. There are no good solutions for 1) as the specification precisely aims at discovering devices. For 2), the problem is that discovered devices may not have been designed with Web interaction in mind. The specification eventually required these devices to implement CORS [CORS], de facto excluding support for legacy devices. Developments on the Network Service Discovery specification have stalled since February 2014 with no plan to revive this work as MediaScape draws to a close. The on-going Presentation API specification [PRES] that MediaScape support from its inception, proposes an alternative approach to discovering other devices. See section 3.2 below for details.

The Network Web Sockets proposal has not been endorsed by the Device APIs Working Group either. Although MediaScape does not plan to push for its adoption, the project notes that the proposal proved useful to implement the concept of "MediaScape-aware" devices that can be discover each other on a local network.

2.1.1.2. HTML Working Group

The HTML Working Group[3] is responsible for the evolution of the HTML language and published the HTML5 specification [HTML5] as a final Web standard in October 2014.

From a MediaScape perspective, media-related features of the specification addressed by the Media Task Force within the Working Group is a key target for bug reports and requests for enhancements that work on cross-device content synchronisation in WP4 revealed, in particular related to the variations of support for the playbackrate property of media elements, and the possibility to extend media elements to support external media timelines.

2.1.1.3. CSS Working Group

As its name suggests, the CSS Working Group[4] develops CSS specifications. Over the years, the group has introduced a series of layout specifications that MediaScape reviewed when considering the best options available to adapt the user interface of an application that runs on multiple screens at once:

- *block layout*, designed for laying out documents, defined in CSS 2.1 [CSS21]
- *inline layout*, designed for laying out text, defined in CSS 2.1
- *table layout*, designed for laying out 2D data in a tabular format, defined in CSS 2.1
- *positioned layout*, designed for very explicit positioning without much regard for other elements in the document, defined in CSS 2.1
- *flex layout*, designed for laying out more complex applications and webpages, defined in CSS Flexible Box Layout Module Level 1 [FLEX]
- *grid layout*, designed for laying out content into arbitrary slots in a flexible or fixed predefined layout grid, defined in CSS Grid Layout Module Level 1 [GRID]

Among these layouts, the grid layout is optimized for user interface design, making it a very good starting point to address cross-device UI adaptation concerns (see deliverable D5.3 Final Version of multi-device adaptation).

[3] HTML Working Group home page: http://www.w3.org/html/wg/

[4] CSS Working Group home page: http://www.w3.org/Style/CSS/members

2.1.1.4. WebApps Working Group

The WebApps Working Group[5] develops core client-side APIs for the Web. MediaScape was initially particularly interested by the Indexed Database API specification that could have inspired the multi-device data sharing task in WP4, and could perhaps be extended for multi-devices scenarios.

MediaScape partners eventually decided against pursuing these directions from a standardisation perspective. The notion of shared state defined in WP4 should more be viewed as a programming model than as a specific set of interfaces (see section 2.2.2).

2.1.1.5. Web and TV Interest Group

The Web and TV Interest Group[6] was created in 2011 as a pre-standardisation forum within W3C for Web and TV companies. The Interest Group studies the convergence of Web and TV and develops use cases and requirements for possible missing blocks. The group has been discussing second screen scenarios and scenarios for cross-device media synchronisation that are in scope for MediaScape in WP3 and WP4.

More broadly speaking, the Web and TV IG is a very good forum to disseminate ideas that are directly related to the convergence of Web and TV. MediaScape used the IG to disseminate cross-device synchronization ideas developed by WP4, push for the creation of the Multi-Device Timing Community Group, and report on progress made on the resulting Timing Object specification [TIMING].

2.1.1.6. Second Screen Presentation Working Group

The Second Screen Presentation Working Group was created in October 2014 within W3C to standardise the Presentation API specification, that allows a web application to request display of web content on a connected display, with a means to communicate with and control the web content from the initiating page. The scope of the Presentation API is relevant to WP3.

MediaScape directly supported the creation and activities of this Working Group through the organisation of a standardisation workshop (see section 3.1), contributions to the chartering phase of the group, and support of the participation of MediaScape partners BBC and W3C in the group (see section 3.2).

2.1.1.7. Multi-Device Timing Community Group

The Multi-Device Timing Community Group was created in February 2015 within W3C by MediaScape partners with support from the Web and TV IG to disseminate the outcomes of WP4 on cross-device synchronization and develop the Timing Object specification [TIMING] to expose timing to web applications and enable cross-device synchronization from within web browsers (see section 3.3).

2.1.2. HbbTV / ETSI

The HbbTV Association[7] develops the HbbTV platform, an iTV platform mainly deployed in Europe. HbbTV references a number of Web, DVB and Open IPTV Forum specifications. Specifications produced by the HbbTV Association become official standards at ETSI[8].

The HbbTV Association published the HbbTV 2.0 specification [HBBTV] in February 2015. This specification includes second screen facilities that allow an HbbTV set to launch applications on a companion device, and a compatible companion device to launch applications on an HbbTV set. Both scenarios are of interest for MediaScape, which has a strong focus on broadcasters. Most second-screen scenarios involve the user's TV set.

2.1.3. RadioDNS/WorldDMB/ETSI

RadioDNS is an organisation which develops specifications around the hybrid radio idea. Currently specified by RadioDNS and recently published in cooperation with WorldDMB are the following specifications/standards:

[5] WebApps Working Group home page: http://www.w3.org/2008/webapps/

[6] Web and TV Interest Group home page: http://www.w3.org/2011/webtv/

[7] HbbTV Association: https://www.hbbtv.org/

[8] ETSI: http://www.etsi.org/

- TS 103 270 v1.2.1 – RadioDNS Hybrid Radio; Hybrid lookup for radio services [TS103]
- TS 101 499 v3.1.1 – Slideshow [TS101]
- TS 102 818 v3.1.1 – Service and Programme Information [TS102]

RadioWEB, an on-going task within RadioDNS, develops a specification which is mainly the adaptation of the HbbTV use case into the radio world.

2.1.4. EBU / ETSI

The EBU[9] publishes specifications for broadcasters. EBU does not have directly relevant activities for MediaScape when the project is started but is a relevant target for the outcomes of MediaScape that may be more specific to TV and radio broadcasters. The authentication work within MediaScape is being done as part of the EBU Cross Platform Authentication group, a subgroup of the EBU's TV Platforms and Services strategic programme. MediaScape supported the work of BBC in that group and the resulting standardisation at ETSI.

2.2. Candidate topics and contributions

This section lists the initial candidate topics for standardisation, established at the beginning of the project from the expected outcomes of the technical tasks (WP3, WP4, and WP5), along with adjustments based on the actual outcomes of these tasks. The descriptions of the tasks below are taken from the DoW.

2.2.1. WP3 – Multi-connection and authentication through multiple devices

The goals of this work package was threefold: determine a mechanism to discover resources such as the different devices, users, services and content available (Task 3.1), determine a solution for dynamic pairing of the resources for a specific session (Task 3.2) and provide a single coherent solution to authenticating users to an internet-provided service on IP-connected media devices from restricted input/output devices such as hybrid radios with push button input and LEDs, to Smart TVs with paired smartphones (Task 3.3).

Research conducted in the field (see D3.3 Final Version of multi-connection mechanisms and multi-device authentication) identified a number of protocols and mechanisms that may be used for discovery and pairing of devices, including mDNS, UPnP, DIAL, IRT's HbbTV Second Screen Framework, NFC, Bluetooth, Google's Physical Web, Network Web Sockets, the Network Service Discovery API, etc. While some of these mechanisms overlap, the right mechanism to use depends on the initial context. For instance, DIAL [DIAL] is useful to discover a device and launch an application on that device, but cannot be used to pair instances of some Web application that is already running on two devices. Similarly, the mechanism to use depends on whether the user is authenticated or not and whether devices are online and already in communication with a back-end server.

Web browsers do not yet expose discovery and pairing mechanisms, despite on-going efforts at W3C when the project started:

- The **Network Service Discovery API** specification, whose development has stopped due to its lack of support for legacy devices and lack of interest from implementers (as seen in section 2.1.1.1)
- The **Network Web Sockets** proposal, which proved useful to develop prototypes in MediaScape but does not have any official standardisation standing.
- An NFC Working Group[10] was created early 2014 at W3C to develop an **NFC API**. Initially, the API required a privileged runtime. Participants eventually agreed that this was the wrong approach, and the Working Group was closed as a result. However, led by Intel, some of them decided to restart the work in a W3C Community Group, scoping down the API to operations that may be achieved within a regular Web runtime without jeopardizing the user privacy and security. Intel demonstrated an early implementation of the API on Chromium at the W3C Technical Plenary and Advisory Committee (TPAC) meeting in Sapporo, Japan, end of October 2015.
- A Bluetooth Community Group[11] was created in August 2014 at W3C to develop a **Bluetooth Low-Energy API** [BLUE]. A first implementation, developed by Google, is available in Chromium. The Community Group should transition to a Working Group once the specification has stabilized.

MediaScape developed prototypes in WP6 to explore multi-device bootstrapping issues and prioritize

[9] EBU: http://www3.ebu.ch/home

[10] NFC Working Group homepage: http://www.w3.org/2012/nfc/

[11] Web Bluetooth Community Group: https://www.w3.org/community/web-bluetooth/

contributions to these standardisation activities.

One specific starting context has proved of particular interest in both internal and external discussions: a user is at home, runs a Web application on a device, e.g. a mobile phone, and wants the application to launch and make use of another device, e.g. a TV set, listening on the same local area network. In particular, this second-screen scenario was raised during the Fourth W3C Web and TV Workshop (see section 3.1) organized by MediaScape. Following the support received during that workshop, the project actively contributed to the creation of the **Second Screen Presentation Working Group**[12] at W3C, and to the Presentation API specification [PRES] that this group develops to enable this scenario (see section 3.2).

The Presentation API remains agnostic of the underlying discovery and launch protocol. Together with the GLOBAL ITV EU project, MediaScape investigated the possibility to align the Presentation with DIAL as used in HbbTV 2.0.

On the authentication front, the project identified a lack of standard way to authenticate users on devices with restricted inputs/outputs mechanisms. The BBC proposed and developed the **Cross Platform Authentication protocol [CPA]**, based on previous related work in RadioDNS[13] and EBU MediaAuth, for Task 3.3. The development took place within an EBU project group chaired by the BBC, leading to the publication of the protocol as an EBU recommendation in September 2014 and should be published as an ETSI standard in 2016 (see section 3.4).

2.2.2. WP4 – Synchronisation

The goals of this Work Package were to develop solutions that enable multi-device data sharing (Task 4.1), contextual state management (Task 4.2) and cross-device content synchronisation (Task 4.3). Task 4.1 and Task 4.2 focused on consistent data sharing of highly dynamic data, whereas Task 4.3 focused on a variety of challenges related to timing and time-based synchronisation of continuous and non-continuous linear data.

Deliverable D4.1 Architecture design of Multi-device Synchronisation Services describes this overall architecture with three related blocks: Shared State, Shared Context and Shared Motion.

Shared State is much more a programming model than a specific interface that would directly benefit from standardisation. For instance, the implementation used in MediaScape uses a simple key-value store but the Shared State programming model is not limited to keys and values. The main ideas behind Shared State are that:

1. Multi-device application authors must acknowledge the inherent asynchronous nature of write operations, required to achieve consistency and efficiency, and keep a clean separation between read and write operations in the code. Typically, applications should only react to state updates when the corresponding change event is received.
2. When read and write operations are cleanly separated in the code, many cross-device state synchronisation issues simply disappear.

The semantics of **Shared State** are analogous to the paradigms used Facebook's Flux[14] application architecture for building user interfaces (unidirectional data flow, determinism, local reads and sequential writes through dispatched actions).

Similarly, **Shared Context** is an implementation of a common Publish-Subscribe pattern on top of a stateful store (Shared State in the case of MediaScape). Here as well, other implementations are possible and the interface itself is not necessarily a good candidate topic for standardisation.

Shared Motion provides a cross-device content synchronisation mechanism that builds upon the notion of timing resources. Although media synchronisation across devices has been raised as an important topic in the past, e.g. within the Web and TV Interest Group at W3C, there were no on-going standardisation efforts at the Web layer when the project was started and no identified solution, which maked Shared Motion a good candidate for standardisation.

Norut worked with W3C to gather support among member companies involved in the Web and TV IG at

[12] Second Screen Presentation Working Group home page: https://www.w3.org/2014/secondscreen/

[13] RadioDNS: http://radiodns.org/

[14] Facebook Flux: https://facebook.github.io/flux/

W3C, leading to the creation of the Multi-device Timing Community Group[15] and the development of the Timing Object specification (see section 3.3).

2.2.3. WP5 – Multi-device Adaptation

What happens when a new device joins a multi-device session? If the application plays a video, it may want to move the video to a larger screen when one becomes available, or it may move textual content or interactive graphics to the user's tablet for convenience, etc. Possible combinations grow exponentially with the number of devices and possible roles involved. The code needed to handle these different combinations is tedious, hard to maintain and hard to debug. Also, usual responsive design patterns may not be enough to cover multi-device scenarios as the user interface on a given device also depends on the presence of other devices in the session. This work package designed mechanisms and flexible APIs (Task 5.3) to enable broadcasters and application developers to create one single application code that allows a cross-device application behaviour (Task 5.1), and provides user interface adaptation mechanisms for the split up of the elements and functionality of an app over multiple devices (Task 5.2).

The underlying goal was to simplify the development of multi-device web applications. Balancing abstraction and expressiveness is hard in practice and it was deemed premature to propose standardisation for a complex topic that would sill benefit from additional research.

Standardisation areas that were investigated included:

1. A vocabulary to describe the role and features of the different items in the application to be used by the adaptation toolkit depending the multi-device context. The project noted previous attempts were made to work on core vocabularies at W3C, notably the Device Description Repository Core Vocabulary [DDR] and the Delivery Context Ontology [DCO].
2. A declarative format for multi-device adaptation rules based on a combination of explicit and implicit rules described in D5.3 Final Version of multi-device adaptation, where explicit rules allow developers to define a specific action over a specific context and implicit rules enable developers to define some hints to the application elements and each device will dynamically calculate the best adaptation outcome depending the current context. More hands-on experience with these rules is needed though for this format to be a good candidate topic for standardisation.
3. Enhancements to CSS to simplify the authoring of responsive multi-device user interface.

Instead of a vocabulary, the project could also contribute on defining best-practices to take into account when creating a multi-device media application and defining a list of properties that are not yet exposed to Web applications but that would be useful to enable specific scenarios. For instance, there is no easy way for a Web application to detect the actual physical screen size of a device. Workarounds rely on the resolution and the pixel density to approximate the actual size. Knowing the actual size of the screen is important in multi-device applications to have a video play on the largest device by default (which may not be the device with the highest screen resolution). That said, exposing the role of the device may be more important than exposing the actual screen size. As said, the project needs to identify the exact needs here.

Enhancements to CSS may take different forms. Through testing, Vicomtech-IK4 identified four relevant parameters that affect the layout to use in a multi-device application: the type of device, the number of components that compose the application, the nature of the application and the number of devices used concurrently (see the *UI Adaptation prototype* in D6.2 Initial Testing Report). The ability to define custom CSS properties for components, common for different types of layouts, proved useful in these tests. Tests also revealed that building a good grid layout is hard in the generic case, especially because CSS Media Queries are useful to have an element react to changes in the viewport size but cannot be used to have an element react to changes in the size of their parent container. The project notes initial discussions on Use Cases and Requirements for Element Queries [RICG] within the Responsive Issues Community Group[16] at W3C. In both cases, project partners need to investigate the topic further before they can make useful contributions to standards.

[15] Multi-device Timing Community Group home page: http://www.w3.org/community/webtiming/

[16] Responsive Issues Community Group home page: http://www.w3.org/community/respimg/

3. Contributions to standardisation activities

This section describes standardisation contributions that MediaScape made during the first 17 months of the project. The section will be updated in the final Standardisation Report to include the activity during the rest of the project's lifetime.

3.1. W3C Standardisation workshop

3.1.1. Context

When the project started, W3C had already run a series of three workshops dedicated to the convergence of the Web and TV, in 2010 and 2011 in Tokyo, Berlin, and Hollywood. These workshops have proved useful to share proposals, set priorities, and gather the initial support needed to start actual standardisation efforts within W3C. Works on the Network Service Discovery API, Media Source Extensions (to enable adaptive streaming and time shifting use cases) and Encrypted Media Extensions specification (DRM support) are examples of efforts that received their initial impulse during one of these Workshops.

Building on the success of the first three workshops, MediaScape supported the organization of a **Fourth W3C Web and TV workshop**[17] in Munich, hosted by IRT, on 12-13 March 2014, with a strong focus on Second- and multiple screen scenarios.

3.1.2. Call for papers and agenda

The workshop was chaired by:

- Yosuke Funahashi from TomoDigi
- Ralf Neudel from IRT
- Giuseppe Pascale from Opera Software
- Mark Vickers from Comcast

The call for papers singled out three areas for new developments:

1. **Second- and multiple screen scenarios**, including ways to control multiple screens, share data across devices, and author Web content for multiple devices
2. **Hybrid TV**, including migration to HTML5 and the identification of potential gaps in HTML5 to enable the next-generation of broadcasting
3. **Social television**, including discussions on integration of TV and social networks

The first area is directly aligned with the developments conducted in MediaScape on purpose. The second and third areas are in scope for MediaScape as well. In particular, MediaScape builds on Web-based technologies and thus needs to encourage the adoption of HTML5 and migration to it within iTV systems. The project also detailed social TV scenarios in WP2 (see D2.1 Usage Scenarios and Requirements).

Roughly 30 papers were submitted for the workshop, the majority of which concentrated on the multi-screen applications and hybrid TV areas. Norut, NEC, W3C and Vicomtech submitted a paper for MediaScape entitled “An architecture for second screen experiences based upon distributed social networks of people, devices and programs” [MSWS]. The paper was accepted and Dave Raggett from W3C presented it during the workshop.

[17] Fourth W3C Web and TV Workshop: http://www.w3.org/2013/10/tv-workshop/

3.1.3. Outcomes

Figure 1 Participants in the Fourth W3C Web and TV Workshop

There were 126 participants from 88 organizations, both W3C members and non-members: Abertis Telecom, Alcatel-Lucent, Apple, ARD, ARM, AT&T, Bayerischer Rundfunk, BBC, BLM (Bayerische Landeszentrale für neue Medien), BSkyB, C-DAC (Centre for Development of Advanced Computing), CableLabs, Cisco, Conax, Conbox, Condat, CreateCtrl, Czech Television, Deutsche Telekom, DFKI, Digital TV Labs, DIT (Deggendorf Institute of Technology), Dolby, Dong-Eui University, EBU (European Broadcasting Union), EchoStar, Elgato Systems, Ericsson, Espial, ETRI (Electronic and Telecommunications Research Institute), France Télévisions, Fraunhofer FOKUS, Freesat, Fuji Television Network, Huawei Technologies, Igalia, Irdeto, IRT, Japan Ministry of Internal Affairs and Communications, Kaonmedia, KDDI, LG Electronics, LTFE, Luxunda, M-net Telekommunikations, MDR (Mitteldeutscher Rundfunk), MNITI, MODUL University, MPAA, MStar Semiconductor, NBCUniversal, NDR (Norddeutscher Rundfunk), NHK (Japan Broadcasting Corporation), Nippon TV, NPO (Netherlands Public Broadcasting), NTT (Nippon Telegraph and Telephone Corporation), Opera, Orange, Oriental Cable Network, Panasonic AVC Networks, Ph.D Student, Polytechnic University of Bari, RAI Research Center, RTL Group, RTV Slovenia, Samsung Electronics, SES Platform Services, Sky Deutschland, Sky Italy, SmarDTV, Sony Corporation, St. Pölten University of Applied Sciences, Strategy & Technology, Swisscom, TBS (Tokyo Broadcasting System), Télécom ParisTech, Thomson Video Networks, TNO (Netherlands Organization for Applied Scientific Research), Tomo-Digi, Toshiba Corporation, Toshiba Information Systems UK, TP Vision, TV Asahi, TV TOKYO, University of Almeria, W3C, WDR (Westdeutscher Rundfunk), WOWOW.

The workshop's report[18] lists 8 priority topics raised during the workshop:

1. **synchronisation of video and (meta) data, video and audio**
2. Testing devices based on Web technologies
3. Rendering and control of linear video using <video> (so-called «Tuner API»)
4. Misc gaps around delivery & rendering of IP video, mostly integration issues for TV devices
5. **Discovery and communication between two user agents or a user agent and another device/service**
6. Performance measurement (benchmarks) for web technologies/animations
7. Accessibility features
8. Pluggable CDM for EME

MediaScape requires or develops solutions for the discovery and communication between two user agents (or between a user agent and another device/service) in WP3, as well as for synchronisation of video and (meta) data across devices in WP4.

[18]Fourth W3C Web and TV Workshop report: http://www.w3.org/2013/10/tv-workshop/report.html

3.2. W3C Second Screen Presentation Working Group

3.2.1. Context

One of the key outcomes of the workshop is the need to enable discovery and communication between two user agents or between a user agent and another device/service from within a Web browser, a core issue addressed by WP3 in MediaScape.

The Network Service Discovery API, published by the Device APIs Working Group in W3C, was to expose discovery mechanisms directly to Web applications. However, to overcome security issues, the specification required CORS support in devices/services that may be discovered, which de facto made the API incompatible with the large base of legacy devices. The effort stalled as a result: the API is stable but not implemented, with no expressed intent to implement it among browser vendors.

Intel presented a draft of a Presentation API specification during the Workshop. This API takes a different approach to discovery, restricting use cases to discovery and communication between screens and hiding the underlying discovery away from the Web application. The initial proposal was restricted to screens attached through an HDMI connection or a Miracast dongle.

Given the feedback and support received during the Workshop, W3C initiated discussions with interested parties (mainly Fraunhofer FOKUS, Google, Intel, Mozilla, and Netflix) to transition the API to a W3C Working Group so that the specification be endorsed by the W3C Membership and eventually be published as a Web standard.

In the meantime, the API continued to be updated within the Second Screen Presentation Community Group, created by Intel, to add support to other types of second screens such as Google Chromecast devices. Community Groups within W3C are not formally endorsed by the W3C Membership and may be created by anyone. While Community Groups may develop technical specifications, these specifications are not official W3C standards.

3.2.2. Creation of the Working Group

As opposed to Community Groups, Working Groups need to be formally approved by the W3C Membership before they are created. Typically, a draft Working Group charter is assembled and a call for review is sent to representatives of all W3C Members. The draft charter is often updated in the process to address members feedback. The Working Group is created given enough support and no objection raised during the review.

W3C took an active role in the preparation of the draft charter (see Annex 1). The draft charter sets the scope and mission of the Working Group, and defines the deliverables of the group as well as the dependencies and liaisons with other groups and organizations.

The mission of the Second Screen Presentation Working Group is to define a **Presentation API** that allows a web application to request display of web content on a connected display, with a means to communicate with and control the web content from the initiating page and other authorized pages. The API will hide the details of the underlying connection technologies and use familiar, common web technologies for messaging between the authorized pages and the web content shown on the secondary display. In particular, the API is agnostic with regard to the display connection used, and works equally well with display connections that support video only, for example, a TV connected to a laptop with an HDMI connection or a Miracast dongle, and with remote devices such as Google Chromecast or DIAL devices.

Twenty-five W3C members supported the creation of the Working Group, including Apple, BBC, CableLabs, Google, Fraunhofer FOKUS, Institut Telecom, Intel, Mozilla, Netflix, Sony. No objections were raised during the call for review. The Second Screen Presentation Working Group was created in October 2014 as a result.

From a MediaScape perspective, the Presentation API fits in WP3, as it enables the discovery and pairing of a secondary device (e.g. a SmartTV) with a primary device, and more importantly allows a Web application that runs on the primary device to launch another Web application on the secondary device and exchange peer-to-peer messages with it afterwards. It is interesting to note that the Presentation API addresses security issues that arise during discovery and pairing by not exposing discovery and pairing at all to the requesting Web application, leaving it up to the Web browser to manage these steps.

3.2.3. Staff contact role within the Working Group

Each Working Group at W3C has a technical staff contact. W3C staff contacts are responsible for ensuring the proper running of the Working Group, ensuring that the group follows its charter and the W3C Process, assisting the Working Group chair in driving the work of the group, attracting the critical parties to participate in the standardisation effort, and disseminating the results of their Working Groups. This task also includes

participation in teleconferences, face-to-face meetings, mailing list exchanges, etc. MediaScape supports the role of François Daoust from W3C as staff contact for the Second Screen Presentation Working Group.

3.2.4. Presentation API polyfill

MediaScape also contributed to the development of the API itself, both through the implementation of a polyfill of the Presentation API and through technical contributions to improve and fix the procedures defined in the Presentation API specification.

A polyfill is a code library that runs in a Web browser which provides functionality that is not yet implemented by the browser. The polyfill of the Presentation API was developed in two phases. During the first phase, the goal was to implement the Presentation API on top of the Google Cast extension in Google Chrome, with a view to providing concrete feedback to the Second Screen Presentation Working Group. See deliverable D6.2 Initial Testing Report (section 2.6) for details about the implementation of the first version of this polyfill, as well as for issues that it allowed to raise and discuss within the Working Group.

It quickly appeared that MediaScape was approaching discovery and pairing from a different angle than the working group. Many of the mechanisms explored in WP3 are not *discovery*-based, but *invitation*-based. For instance, the Physical Web allows a device to broadcast the URL of an application, thus inviting nearby devices to the session. Named Web sockets and QR codes have been used similarly. The second implementation phase explored the implementation of the Presentation API on top of a couple of invitation-based mechanisms (QR codes and the Physical Web) that among other things do not require or allow the establishment of a peer-to-peer communication channel, again with a view to providing feedback to the Working Group. See deliverable D3.4 MediaScape multi-connection mechanisms and multi-device authentication for details about the implementation of this second version of the Presentation API polyfill.

This work was presented to, and discussed with, the Second Screen Presentation Working Group during a face-to-face meeting in Sapporo, Japan, end of October 2015. The polyfill shows the limits of an approach that mixes invitation-based and discovery-based mechanisms. The expectations in terms of privacy are very different in both cases, in particular. For discovery-based mechanisms, the user agent is responsible for discovery and controlling any remote device on behalf of the user. No information (e.g. the URL of the application) is leaked without the user explicitly approving the pairing with another device. In contrast, for invitation-based mechanisms, the user agent needs to share the URL of the application and cannot screen the devices that may reply to the invitation beforehand. With discovery, the user chooses a device and implicitly agrees to share information with it; with invitation, the user chooses a mechanism but has much less control over the information shared with others and on the boundaries where this sharing takes place (e.g. all Bluetooth devices within a few meters may be able to eavesdrop the session).

As such, the Working Group envisions the establishment of a communication channel by the requesting user agent as a way to further secure the multi-device session, even though this feature negatively impacts the interoperability of the specification in the mid-term because browser vendors will implement their own set of protocols for discovery, pairing, authentication of devices and encryption of communications.

The main outcome of this exploration is that discovery-based and invitation-based mechanisms should probably rather be exposed under different interfaces, so as not to confuse users.

3.2.5. Compatibility with HbbTV 2.0

The MediaScape project contributed to investigate the compatibility of the Presentation API with HbbTV 2.0 in collaboration with the GLOBAL ITV EU project. As things stand, while the Presentation API can easily be built on top of the DIAL protocol used in HbbTV 2.0, the Presentation API won't be able to launch a Web application running on a receiving HbbTV set, because HbbTV does not expose the right interfaces for communication on the receiving end. This situation could change in a revision of the HbbTV standard, provided that Presentation API implementers also agree on a common set of protocols to use to establish a secure communication channel, that the HbbTV specification could reference.

3.2.6. Standardisation status

The Presentation API has stabilized early 2016, and should be published as a W3C Candidate Recommendation by end of June 2016. Early implementations of the Presentation API exist in Google Chrome (under the hoods, the Cast SDK in Google Chrome is now implemented on top of the Presentation API) and Firefox.

The Working Group has also started to work on a media-focused version of the Presentation API, in other words on a specification that allows a Web application to request display of audio or video content (as opposed to

a Web app) on a second screen. The Remote Playback API is actively developed under the impulse of Google and Apple, and addresses some of the requirements explored by the BBC in the radio control prototype (see deliverable D3.4 MediaScape multi-connection mechanisms and multi-device authentication).

3.3. W3C Multi-device Timing Community Group

3.3.1. Context

A second key outcome of the workshop is the need to complete support for synchronisation of video, audio and data in HTML5. HTML5 has partial support for synchronisation through the notion of MediaControllers in particular, but does not address all single device use cases in that field and does not support cross-device synchronisation at all. Besides, as of end of 2015, browser vendors are actually deprecating the MediaController interface, which has not been properly implemented.

MediaScape develops a high-precision, scalable motion synchronisation mechanism called Shared Motion that allows detailed control of media motions, based on an online third party motion provider, in WP4. Norut has gained a lot of experience in the topic in particular (see D4.3 Final implementation of multi-device synchronisation services) and developed prototypes accordingly (see D6.1 Initial Prototypes). Norut and W3C decided to team up to push for the standardisation of Shared Motion interfaces within W3C.

3.3.2. Discussions with the Web and TV IG

In W3C, the Web and TV Interest Group is the central point for discussions around the convergence of Web and TV technologies. W3C Interest Groups do not produce technical specifications but discuss use cases and identify requirements for Working Groups that will develop technical solutions. The Interest Group discussed three use cases related to synchronisation in 2014[19]:

1. **Identical Media Stream Synchronization** where the same media stream plays in sync on separate devices
2. **Related Media Stream Synchronization** where related media streams, e.g. the same scene viewed from different angles, play in sync on a single Web page or on separate devices.
3. **Clean Audio** where a user who has lost some of his hearing plays the Clean Audio track on his mobile phone while watching a movie with friends in a movie theater or in front of a TV.

The group had not identified technical solutions to address these use cases. To help raise the attention of W3C Members that participate in the Web and TV Interest Group, Norut prepared a scientific paper with W3C entitled “MediaElements, TrackElements & MediaControllers – Enabling Multi-device, Web-based, Linear Media with Shared Motion” [MEDIA] that describes the principles of Shared Motion and outlines a proposal to update HTML5 to enable cross-device synchronisation scenarios. The paper was shared with the Web and TV Interest Group in December 2014, along with demos for each of the use cases identified by the Interest Group.

Interest Groups at W3C cannot produce technical specifications. Further discussions with the Web and TV Interest Group chairs led to the conclusion that a draft technical specification should be assembled in a separate Community Group before it may be brought to a Working Group, likely the Media task force of the HTML Working Group, that addresses media related issues in HTML.

While Community Groups are not formally endorsed by the W3C Membership, they provide a good forum for technical discussions with a clear IPR regime. In practice, more and more specifications developed by Working Groups originate from preliminary work in Community Groups.

3.3.3. Creation of the Community Group

It is good practice to define a precise charter for the Community Group that describes its mission, goals and scope. Norut and W3C prepared this charter (see Annex 2). The Community Group was set to:

- Define a common API for multi-device timing resources, i.e, clocks, stop-watches, timeouts and controllers.
- Review existing programming concepts for timed operation, i.e., setTimeout, setInterval, HTML5MediaElement, HTMLTrackElement, WebAnimation, and suggest adjustments to simplify integration with multi-device timing resources.

[19]List of use cases for 2014 in the Web and TV IG: http://www.w3.org/2011/webtv/wiki/New_Ideas

- Draft a protocol specification for a common interface between Web browsers and providers of multi-device timing resources.
- Discuss native support for multi-device timing in Web browsers.

The outcome of these activities is a combination of bug reports, API proposals and protocol specifications. The charter lists a series of topics that are out of scope to further clarify the scope of the Community Group. Timing mechanisms based on assumptions of synchronized system clocks such as NTP are out of scope for instance, as synchronized system clocks is not a realistic assumption in the Web environment.

3.3.4. Timing Object specification

Once the Community Group was launched, Norut and W3C worked within this group on a draft proposal for an API that Web browsers could implement to expose cross-device synchronization primitives to Web applications. The resulting specification, called **Timing Object [TIMING]**, builds on the research done in WP4. On top of project partners, Community Group participants include representatives from CWI, ERICSSON, Fraunhofer FOKUS, INRIA, Institut Telecom, Intel, KDDI, and Orange.

The timing object is a local object that may be used by Web clients to ensure precisely timed operation as well as flexible timing control. If multiple timing-sensitive components take direction from the same timing object, their behaviour will be precisely aligned in time (synchronized). Crucially, this is also the case in distributed settings. A central motivation for the timing object is that it may be connected to an online timing resource. This way, the local timing object is a gateway to precisely timed operations, both in single-device and multi-device scenarios.

The specification describes how media playback in HTML5 would need to be updated to be able to connect it to an external timeline. It also includes the notion of sequencer (developed in WP4) for timed data. Alignment with other types of timed data (e.g. content produced with the Web Audio or with Web Animations) should be doable, although not included in the draft specification.

3.3.5. The timingsrc library

Norut developed a pure JavaScript implementation of the Timing Object specification in the **timingsrc** library, available on GitHub[20]. The timingsrc library features programming concepts related to the Timing Object specification. In particular, the MediaSync library and the Sequencer concept developed for shared motion in WP4 were adapted to use the Timing Object API instead of the MediaScape specific Shared Motion API. Shared Motion is exposed in the timingsrc library as an online timing provider that may be associated with a timing object. The timingsrc library also includes additional programming tools for precise timing.

By providing a basic toolset for use with timing objects, timingsrc essentially provides a new programming model for precise timing, synchronization and control in single-page as well as multi-device Web applications.

Norut also developed a Website for timingsrc[21] where Web developers can find live demonstrations, code examples, as well as rich documentation of the API and of background considerations that led to it.

3.3.6. Shared demos and MediaSync reports

Norut developed a number of demos that showcase the cross-device (and cross-browser) media synchronization that may be achieved today, along with limits that current browser have. These demos were posted on the Multi-Device Timing Community Group blog:

Date	Title	URL
5 Feb. 2015	Media Sync: Carnival - Chrome and Firefox side-by-side	https://www.w3.org/community/webtiming/2015/02/05/welcome-multi-device-timing/
11 Mar. 2015	Media Sync: Timed multi-device internet radio - synchronisation of live radio with Shared Motion	https://www.w3.org/community/webtiming/2015/03/11/35/

[20] Timingsrc open-source library: https://github.com/webtiming/timingsrc

[21] Timingsrc Web site: http://webtiming.github.io/timingsrc/

Date	Title	URL
27 Mar. 2015	Media Sync: Synchronising across mobile platforms	https://www.w3.org/community/webtiming/2015/03/27/the-motion-corp-band/
8 Apr. 2015	MediaSync + Sequencer: Multi-device linear media	https://www.w3.org/community/webtiming/2015/04/08/timed-data-and-multi-device-playback/
17 Aug. 2015	MediaSync: Synchronizing MPEG-DASH - Shaka player	https://www.w3.org/community/webtiming/2015/08/17/multi-device-mpeg-dash/
2 Mar. 2016	MediaSync: Synchronizing 30 videos in a single web page	https://www.w3.org/community/webtiming/2016/03/02/inter-device-sync/

Table 1 Demos for the Multi-Device Timing CG

In parallel, Norut worked on two reports for the group:

1. **Distributed synchronization of HTML5 media** [NORDS], which analyzes the quality of synchronization to be expected when synchronizing HTML5 audio and video on multiple devices using Shared Motion.
2. **MediaSync Report 2015** [NORMS], which provides an extensive analysis of timing aspects of HTML5 media across browsers, operating systems and media formats, including how players repond to time-shifting and adjustments in playback-rate.

Experiments with MediaSync revealed issues with precise synchronisation for some particular combinations of architecture, browser and media type. These issues mostly relate to poor implementation of certain properties of the media element, but ambiguity in standards may have a role to play as well. In particular, chrome on Android lacks support for variable playback rate. This is being sorted out with the introduction of the Uniform Media Pipeline rolling out when this report is written. Another issue related to the approximated value for *currentTime*, and how internal delays are accounted for. Some mobile devices running Chrome on Android were synchronized correctly with respect to currentTime, and yet the audio was consistently delayed relative to reference devices. This indicated that some internal delay had not been accounted for.

Norut raised this issue on the Chrome development forum[22]. It turned out that differences in buffer size made for variances in delay between devices. The issue was easily solved by having *currentTime* updated more often and also compensate for buffering delay[23]. As of March 2016, this bugfix is already in developer versions and will be available in standard android withing a few months.

This issue can not be detected in single-device media playback, but is problematic in multi-device playback. As such, this was an issue that the media team at Google Chrome was not aware of. A MediaScape demo was used by the Chrome team to verify the issue and presumably evaluate the bug fix.

3.3.7. Assessment of support for the creation of a Working Group

MediaScape investigated the possible transition of the Timing Object specification to a Working Group that could refine it and publish it as a Web standard. Support from a few W3C members is required for such a transition to take place. W3C presented the Timing Object specification, along with a series of demos prepared by Norut, to the Web and TV Interest Group[24], the Digital Signage Business Group, as well as to interested participants from in a breakout session[25], at the W3C TPAC event in Sapporo, end of October 2015.

Cross-device synchronization has also been flagged at W3C as a headlight for 2016, and presented as such to W3C Advisory Committee in Boston, end of March 2016.

In parallel, Norut and W3C held a number of discussions with various companies (including Fraunhofer

[22]Chromium bug contribution: https://bugs.chromium.org/p/chromium/issues/detail?id=263654#c56
[23]Chromium bug solution: https://bugs.chromium.org/p/chromium/issues/detail?id=263654#c56
[24]Minutes of Web and TV IG F2F: https://www.w3.org/2015/10/25-webtv-minutes.html#item16
[25]Breakout session: https://www.w3.org/wiki/TPAC/2015#Cross-device_synchronization

FOKUS, Google, Intel, IRCAM, Mozilla, the National Association of Broadcasters, Sony, TNO).

Many companies reported interest in the topic. At the same time, they envision the topic from their own perspective, and are more supportive of ad-hoc improvements to different Web specifications and to implementations, than of a generic solution such as the Timing Object specification in this space. As of March 2016, more incubation within the Multi-Device Timing Community Group seems to be necessary before the specification may be standardized. Project partners will continue to lobby in favour of the Timing Object solution, needed to orchestrate arbitrary multi-media content experiences across devices. A face-to-face meeting of the Multi-Device Timing Community Group is planned at the upcoming W3C Technical Plenary and Advisory Committee event in Lisbon, Portugal, end of September 2016.

3.4. EBU Cross-Platform Authentication

3.4.1. Context

As part of research on WP3, the BBC identified the need for an authentication protocol suitable for media devices, such as radios and TVs. Radios typically have small display screens, often limited to two rows of alphanumeric characters, and both radios and TVs are usually controlled using an infra-red remote control, which is difficult to use for text entry. These limitations mean that existing authentication protocols, such as OAuth 2.0 [OAUTH], were not applicable to these kinds of devices. Although early drafts of OAuth 2.0 included a device flow, this was removed from the specification in 2010 due to lack of implementation experience. The OAuth 2.0 Device Profile was later published as a separate draft [ODF], but was never standardised. The lack of an established standard led to the creation of the Cross Platform Authentication group within EBU, chaired by the BBC. There, the BBC proposed and developed the **Cross Platform Authentication protocol [CPA]**, based on experience gained through Task 3.3 and on previous related work in RadioDNS[26] and EBU MediaAuth.

The first phase was dedicated to gathering members and setting the scope of the project. TVP, RTS-SRG, VRT, Global Radio and Frontier Silicon joined the group and contributed by each studying in depth an aspect of the future system, analysing and identifying the gaps in the available technologies for media devices.

In the implementation phase, the BBC and EBU jointly worked on a first draft version of the protocol together with an open source reference implementation. Thanks to this reference implementation, Frontier Silicon was able to very quickly build a proof of concept on a Revo Axis radio. This was demonstrated at the EBU's RadioHack workshop (Geneva, February 2014).

The protocol was published as an EBU recommendation on September 2014 and was demonstrated at the International Broadcasting Convention (IBC) 2014. The BBC then adapted the protocol to work with connected TV, in particular to support HbbTV 1.5 and HbbTV 2.0 compatible devices. The resulting specification was submitted to ETSI for standardisation in November 2015, and should be published as an ETSI standard during April 2016.

3.4.2. Scope of the document

Cross Platform Authentication is a protocol based on OAuth 2.0 which defines how to associate IP connected media devices (e.g., TVs and radios) to an authenticated online account so those devices can interact with broadcaster-provided personalised services.

CPA supports two modes of operation: client mode, in which personalisation is only available on the client device, and user mode, where personalisation is enabled across all of the user's devices. This allows the delivery of personalised services to devices without requiring any user interaction. If a device is registered in client mode, the user can later sign in and associate the device with their online account.

The CPA protocol also defines a standard way for media services to share a common authentication system in order to provide users with a single sign-on experience across different services and devices. It works with any identity provider (CAS, SAML, OpenID, etc).

The protocol can be implemented on a wide range of IP-connected media devices taking into account the range of capabilities of displays and input methods, for instance: set top boxes with infra-red remote controls, and radios with small LCD displays.

3.4.3. Further work

In February 2016, the OAuth Working Group at IETF restarted activity on the OAuth 2.0 Device Profile, with participants from Google, ForgeRock, Microsoft, and ARM. The EBU CPA group has initiated discussion with the

[26] RadioDNS: http://radiodns.org/

IETF group about the possibility of aligning their approaches.

3.5. RadioWEB specification

3.5.1. Context

As seen in section 2.1.3, RadioWEB is an on-going task within RadioDNS, started in June 2014 and led by IRT, that looks into the integration of HTML content alongside broadcast radio. The goal is to combine HTML, CSS, JavaScript and other Web technologies into the RadioDNS hybrid radio standard to add new functionalities for broadcasters.

3.5.2. RadioWEB specification

In 2015, IRT proposed **a first draft of the RadioWEB specification**[27] to the RadioWEB developer group. The scope of this document is subdivided into the following main topics:

- Section 1-3 “Application signalling and discovery”: This part specifies how RadioWEB applications are signalled to the end user device. If necessary, during the service discovery process, different device options (e.g. display size) can be signalled and negotiated.
- Section 4 “Application Life-Cycle”: This part defines the life-cycle of a RadioWEB application. This includes answering questions such as “when to start the web application?”, “when to stop the web application?” and “Can web applications stay alive over service boundaries while preserving their state?”
- Section 5-6 “Content formats and Application API”: This part will specify the amendments to HTML5/CSS and JavaScript, which will enable the RadioWEB application to utilize the built in radio device in order to select other services, access in band data and in band signalisations. Primary goals of this task are:
  - Enable Web developers to easily develop HTML/CSS/JavaScript applications with the focus on accompanying the selected radio programme with additional data, content and links.
  - The developed APIs must be agnostic of the underlying broadcast technology (FM, DAB, HD, …)
  - Make sure that the developed APIs are aligned with the work coming out of other groups - probably the Universal Smartphone Radio project - so there could be a clear mapping between the JavaScript functions and the corresponding native functions.

In order to concentrate the draft to RadioDNS topics, the RadioDNS group is considering submitting the JavaScript API within the W3C TV Control API Community Group, especially as this Community Group is transitioning the work on the TV Control API specification to a Working Group set to publish it as a Web standard.

[27] First version of the RadioWEB draft specification: https://docs.google.com/viewer?a=v&pid=forums&srcid=MTE3NjAzMjU1NjA4MTEyMTI1ODUBMDQ0NDUxMzE5OTg5OTE4MDcxNTQBVFYwajJhWmZ3QTRKATAuMQEBdjI

4. Updated action plan

Table 2 below describes MediaScape's initial standardisation plan, as it was created during the first year of the project. The status column describes the status at the end of the project. Deviations from and adjustments to the plan are discussed next. Note the plan focuses on technical contributions to standards and does not mention the organisation of the standardisation workshop.

WP	Action	Who	Status
WP3	Create Second Screen Presentation WG at W3C	W3C	Done, see 3.2.2
WP3	Staff contact for W3C Second Screen Presentation WG	W3C	Done, see 3.2.3
WP3	Explore bootstrapping scenarios and adjust plan accordingly	BBC, IRT, VIC, W3C	Done, see 2nd implementation phase in 3.2.4
WP3	Initiate discussions with HbbTV for a possible inclusion of the Presentation API (or other relevant solution) in a future version of the HbbTV specification.	IRT, W3C	Proved too early, see below
WP3	Initiate discussions with RadioWEB for a possible inclusion of the Presentation API (or other relevant solution) in RadioWEB.	BBC, IRT, W3C	See below
WP3	Create Cross Platform Authentication group at EBU	BBC	Done, see 3.4
WP3	Publish Cross Platform Authentication protocol at EBU	BBC	Done, see 3.4
WP3	Integrate TV requirements in Cross Platform Authentication protocol and publish updated (or dedicated) spec at EBU	BBC	Done, see 3.4
WP3	Submit Cross Platform Authentication protocol to ETSI	BBC	Done, see 3.4.3
WP4	Prepare paper on Shared Motion to gather support in W3C Web and TV IG	NOR, W3C	Done, see 3.3.2
WP4	Create W3C Multi-device Timing CG	NOR, W3C	Done, see 3.3.3
WP4	Draft Shared Motion specifications in CG	NOR, W3C	Done, see 3.3.4
WP4	Submit media-related bug reports to HTML WG	NOR, W3C	Done, see 3.3.6
WP4	Transition CG to WG (provided enough support from W3C Membership)	NOR, W3C	Explored, see 3.3.7
WP5	Explore ECA rules for the adaptation engine and adjust action plan accordingly	NEC	See below
WP5	Precise custom CSS properties and consider contributions to W3C CSS WG	VIC	See below
WP5	Explore CSS Media Queries at the Element level and consider use cases / requirements contributions to the Responsive Issues Community Group at W3C	VIC	See below

Table 2 Standardisation action plan

In WP3, the exploration of different bootstrapping scenarios led project partners to shift focus on discovery and pairing to invitation-based mechanisms for the second phase of the project. This led to the development of the second phase of the Presentation API polyfill that explores such mechanisms and related discussions within the Second Screen Presentation Working Group.

The project investigated the alignment between the Presentation API and HbbTV 2.0, highlighting a few compatibility issues that would prevent HbbTV from directly being to reference this API. The lack of an open and common set of protocols that browser vendors could implement right away for the Presentation API also creates interoperability problems, making it practically impossible for a browser vendor to support an arbitrary set of TV devices. Several HbbTV contributors (including BBC and IRT) now participate in the Second Screen Presentation Working Group to imagine possible solutions for future iterations of the Presentation API.

Similarly, the focus on RadioWEB so far has been on the API to control the radio tuner, that could eventually get merged with the TV Control API proposal developed by the TV Control API Community Group at W3C, which transitioned to a Working Group mid-April 2016. The integration of the Presentation API should be envisioned afterwards.

Research conducted in WP5 on multi-device adaptation has mostly remained at the research phase. Vicomtech-IK4 and W3C reached out to the CSS Working Group and the Responsive Images Community Group at W3C about the CSS Grid Layout specification and the need to enable CSS Media Queries at the element level, possibly restricted to Web Components. This led to useful exchanges with Igalia in particular, an open source consultancy that implements the CSS Grid Layout specification in Webkit. Igalia is part of the Advisory Board of MediaScape. It also revealed that layout systems in CSS such as block layout, flex layout, table layout, or grid layout, are “costly” features that are only introduced in CSS when use cases, requirements, research and incubation unambiguously highlight the need to work on them. Research and prototypes implemented in MediaScape clearly demonstrate the benefits of simplifying the authoring of multi-device applications for developers. They also show that finding the right abstractions e.g. to use a CSS-like declarative approach is difficult and worth investigating further before the solution may be standardised.

At the end of the project, the need to enable CSS Media Queries at the element level has been well documented, various application developers having blogged about use cases and implemented prototypes of possible solutions. This has not yet triggered any proper standardisation action, but the project notes with interest a recent proposal by Google to develop an observer-style API to give components some way to respond to changes in size, called ResizeObserver[28]. This API would in particular allow an application to adjust the content and layout of a component based on the size of its container. Conceptually, a pure CSS-based solution would be better to adjust the layout of a component, but the CSS solution could well be created afterwards to complement the API in the future.

5. Conclusions

MediaScape explored a number of research topics: discovery, pairing of devices, authentication, multi-device data sharing, multi-device contextual state management, cross-device content synchronisation, cross-device synchronization, authoring of Web applications that automatically span to the devices that are available and rules to adjust their user interface dynamically.

Through research and prototyping, MediaScape identified a few technical gaps that would either enable or simply the authoring of cross-device Web applications and that are not currently covered by existing standards. Whenever possible, the project developed technical solutions and initiated standardisation actions. Not all technical solutions are good candidates for standardisation though. For instance, the project believes concepts such as Shared State are good programming models but would not necessarily benefit from standardisation.

MediaScape made the following 5 main contributions to standardisation:

1. **Organization of a Standardisation Workshop** on the convergence between Web and TV with a strong focus on multi-device scenarios in March 2014.
2. **Creation of the Second Screen Presentation Working Group at W3C** and active contributions to the development of the Presentation API specification that will allow a Web page to request display of and control Web content on a secondary display.

[28] https://github.com/WICG/ResizeObserver/blob/master/explainer.md

3. **Creation of the Multi-Device Timing Community Group at W3C** set to gather support around cross-device synchronization solutions developed in WP4.
4. **Development of the Timing Object specification** at W3C to enable cross-device synchronization within web browsers and **development of the timingsrc library**, a reference implementation of the specification in JavaScript.
5. **Development and publication of the Cross Platform Authentication protocol at EBU and ETSI** that allows to authenticate users on devices with restricted input/output mechanisms.

MediaScape partners will continue to contribute to the activities they helped create. They will in particular contribute to the advancement of the Presentation API along the Recommendation track, promote the use of the Cross-Platform Authentication standard (e.g. for a possible inclusion in OAuth), and work with other interested W3C Members to transition the Multi-Device Timing Community Group to a Working Group.

While not addressed at the standardisation level within the project, MediaScape notes that technical solutions that simplify the development of cross-device Web applications would probably benefit from improvements to standards in the long term.

6. References

[BLUE] Web Bluetooth, Draft Community Group Report, 13 October 2015 (See https://webbluetoothcg.github.io/web-bluetooth/)

[CPA] The Cross-Platform Authentication Protocol, EBU Tech 3366, Sean O'Halpin, Chris Needham, Michael Barroco, September 2014 (See https://tech.ebu.ch/docs/tech/tech3366.pdf)

[CORS] Cross-Origin Resource Sharing, W3C Recommendation, Anne van Kesteren, 16 January 2014 (See http://www.w3.org/TR/2014/REC-cors-20140116/)

[CSS21] Cascading Style Sheets Level 2 Revision 1, W3C Recommendation, B. Bos, T. Çelik, I. Hickson, H. Lie, 7 June 2011 (See http://www.w3.org/TR/2011/REC-CSS2-20110607/)

[DCO] Delivery Context Ontology, W3C Working Draft, Rhys Lewis (ed.), 21 December 2007 (See http://www.w3.org/TR/dcontology/)

[DDR] Device Description Repository Core Vocabulary, W3C Working Draft, Andrea Trasatti, Jo Rabin, Rotan Hanrahan (eds.), 14 April 2008 (See http://www.w3.org/TR/2008/NOTE-ddr-core-vocabulary-20080414/)

[DIAL] DIscovery And Launch protocol specification, v1.7.1, Netflix, 19 November 2014 (See http://www.dial-multiscreen.org/dial-protocol-specification/DIAL-2ndScreenProtocol-1.7.1.pdf)

[EME] Encrypted Media Extensions, W3C Working Draft, David Dorwin, Jerry Smith, Mark Watson, Adrian Bateman, 19 October 2014 (See http://www.w3.org/TR/2015/WD-encrypted-media-20151019/)

[FLEX] CSS Flexible Box Layout Module Level 1, W3C Last Call Working Draft, Tab Atkins Jr., fantasai, Rossen Atanassov et al., 14 May 2015 (See http://www.w3.org/TR/2015/WD-css-flexbox-1-20150514/)

[GRID] CSS Grid Layout Module Level 1, W3C Working Draft, Tab Atkins Jr., fantasai, Rossen Atanassov et al., 17 September 2015 (See http://www.w3.org/TR/2015/WD-css-grid-1-20150917/)

[HBBTV] HbbTV 2.0 Specification, HbbTV Association, 2 February 2015 (See https://www.hbbtv.org/pages/about_hbbtv/HbbTV_specification_2_0.pdf)

[HTML5] HTML5, W3C Recommendation, Ian Hickson et al., 28 October 2014 (See http://www.w3.org/TR/2014/REC-html5-20141028/)

[MEDIA] MediaElements, TrackElements & MediaControllers, Enabling Multi-device, Web-based, Linear Media with Shared Motion, Ingar Arntzen, Njål Borch, François Daoust, Dominique Hazaël-Massieux, Scientific Paper, December 2014 (See https://lists.w3.org/Archives/Public/public-web-and-tv/2014Dec/att-0016/MediaElementsTrackElementsMediaControllers.pdf)

[MSE] Media Source Extensions, W3C Candidate Recommendation, Aaron Colwell, Adrian Bateman, Mark Watson, 31 March 2015 (See http://www.w3.org/TR/2015/CR-media-source-20150331/)

[MSWS] An architecture for second screen experiences based upon distributed social networks of people, devices and programs, MediaScape position paper for the Fourth W3C Web and TV Workshop, Njål Borch, Bin Cheng, Dave Raggett, Mikel Zorrilla, March 2014 (See

http://www.w3.org/2013/10/tv-workshop/papers/webtv4_submission_6.pdf)

[NSD] Network Service Discovery, W3C Working Draft, Rich Tibbett, 20 February 2014 (See http://www.w3.org/TR/2014/WD-discovery-api-20140220/)

[NORDS] Distributed Synchronization of HTML5 Media, Njål T. Borch and Ingar M. Arntzen, December 2014 (See http://norut.no/sites/norut.no/files/norut_tromso_rapport_15-2014.pdf)

[NORMS] MediaSync Report 2015: Evaluating timed playback of HTML5 Media, Njål T. Borch and Ingar M. Arntzen, December 2015 (See http://norut.no/sites/norut.no/files/norut_tromso_rapport_28-2015.pdf)

[OAUTH] The OAuth 2.0 Authorization Framework, D. Hardt, Ed., RFC 6749, October 2016 (See https://tools.ietf.org/html/rfc6749)

[ODF] OAuth 2.0 Device Profile, D. Recordon, Ed., July 2010 (See https://tools.ietf.org/html/draft-recordon-oauth-v2-device-00)

[PRES] Presentation API, W3C Working Draft, Mark Foltz, Dominik Röttsches, 13 October 2015 (See http://www.w3.org/TR/2015/WD-presentation-api-20151013/)

[RICG] Use Cases and Requirements for Element Queries, Editor's Draft, W3C Responsive Issues Community Group, 15 January 2015 (See http://responsiveimagescg.github.io/eq-usecases/)

[TIMING] Timing Object, Draft Community Group Report, Ingar M. Arntzen, François Daoust, Njål T. Borch, Multi-Device Timing Community Group, 19 October 2015 (See http://webtiming.github.io/timingobject/)

[TS101] Hybrid Digital Radio (DAB, DRM, RadioDNS); SlideShow; User Application Specification, v3.1.1, ETSI Technical Specification, January 2015 (See http://www.etsi.org/deliver/etsi_ts/101400_101499/101499/03.01.01_60/ts_101499v030101p.pdf)

[TS102] Hybrid Digital Radio (DAB, DRM, RadioDNS); XML Specification for Service and Program Information (SPI), v3.1.1, ETSI Technical Specification, January 2015 (See http://www.etsi.org/deliver/etsi_ts/102800_102899/102818/03.01.01_60/ts_102818v030101p.pdf)

[TS103] RadioDNS Hybrid Radio; Hybrid lookup for radio services, v1.1.1, ETSI Technical Specification, January 2015 (See http://www.etsi.org/deliver/etsi_ts/103200_103299/103270/01.01.01_60/ts_103270v010101p.pdf)

7. Abbreviations and Acronyms

CORS	↔	Cross-Origin Resource Sharing
CG	↔	Community Group
CSS	↔	Cascading Style Sheets
DIAL	↔	DIscovery and Launch
DoW	↔	Description of Work
DRM	↔	Digital Rights Management
DVB	↔	Digital Video Broadcasting
EBU	↔	European Broadcasting Union
ECA	↔	Event Condition Action
ETSI	↔	European Telecommunications Standards Institute
HbbTV	↔	Hybrid Broadcast Broadband TV
HTML	↔	Hypertext Markup Language
IETF	↔	Internet Engineering Task Force
IG	↔	Interest Group
iTV	↔	interactive TV
mDNS	↔	Multicast Domain Name System
NFC	↔	Near Field Communication
SDO	↔	Standards Developing Organisation
UI	↔	User Interface
UPnP	↔	Universal Plug and Play
WG	↔	Working Group
W3C	↔	World Wide Web Consortium

Annex 1: Second Screen Presentation Working Group Charter

This is a copy of the Second Screen Presentation Working Group Charter prepared by W3C in collaboration with other W3C Members. The text has been slightly re-formatted for better readability in this report. The reference Charter is available online at:

http://www.w3.org/2014/secondscreen/charter.html

The mission of the Second Screen Presentation Working Group is to provide specifications that enable web pages to use secondary screens to display web content.

End date: 31 October 2016
Confidentiality: Proceedings are public
Initial Chairs: Anssi Kostiainen
Initial Team Contacts (FTE %: 20): François Daoust
Usual Meeting Schedule:
Teleconferences: topic-specific calls may be held
Face-to-face: we will meet during the W3C's annual Technical Plenary week; other additional F2F meetings may be scheduled (up to 2 per year)
IRC: active participants, particularly editors, regularly use the #webscreens W3C IRC channel

1. Goals

Web content is available on an ever expanding array of devices including ebook readers, phones, tablets, laptops, auto displays, and electronic billboards. These devices have a variety of display screens. There are also a variety of mechanisms that allow these devices to use secondary display screens available in the local environment, attached by wired connections or remotely with wireless, peer-to-peer media.

Common attachment methods include video ports like VGA, DisplayPort or HDMI, or wirelessly through Miracast, WiDi, or AirPlay. Wireless screens may be available on a local area network or over the Internet, brokered by a cloud service. A device like a laptop could provide a screen for a smaller device like a phone.

For many of these techniques the operating system hides how the screen is attached and provides ways for native applications to use the screens. Native applications on an operating system can easily use these additional screens without having to know how they are attached to the device. At this point, however, there is no way for a web page to take advantage of these available secondary displays.

The Second Screen Presentation Working Group aims at defining simple APIs that allow web applications to show and control web content on one or more secondary displays.

2. Scope

The scope of this Working Group is to define an API that allows a web application to request display of web content on a connected display, with a means to communicate with and control the web content from the initiating page and other authorized pages. Pages may become authorized to control the web content by virtue of sharing an origin with the originating page, explicit user permission, or other facilities provided by the user agent. The API will hide the details of the underlying connection technologies and use familiar, common web technologies for messaging between the authorized pages and the web content shown on the secondary display. The web content may comprise HTML documents, web media types such as images, audio, video, or application-specific media, depending on the capabilities of the secondary display for rendering the media. Application-specific media includes that whose type is known to the controlling page and the connected display, but not necessarily a generic HTML User Agent.

The API will provide a means to identify whether requesting display on second screens is likely to be successful, i.e. whether at least one secondary screen is available for display.

The API is agnostic with regard to the display connection used, and also works with display connections that support video only, for example, a TV connected to a laptop with an HDMI connection. In such a usage scenario, the web content displayed on a connected display must be rendered and converted to video before it is

sent to a second screen. The User Agent may use whatever means the operating system provides to prepare and send the video to a second screen. Any interaction between the authorized web pages and the content displayed on a secondary screen would happen within the bounds of the initiating device since both the pages and the content are rendered on that same device, and only a video representation is sent to the second screen.

Alternatively, if the second screen device understands some other means of transmitting content to a display and a means of two-way message passing, the web content can be rendered by the remote device. In this scenario, a URL to the content to be displayed is sent to the secondary display to be rendered there. Because the content is rendered separately from the initiating user agent, pages hosted by other user agents may be authorized to control the remotely rendered content at the same time.

For a requested piece of web content, how and by which device the content is rendered is an implementation detail. The user agent is responsible for determining which secondary displays are compatible with the content that is requested to be shown through the API.

Sending content to a connected display creates a presentation session. Applications can create multiple presentation sessions to control multiple displays, although synchronization between them is not currently supported by the API.

The specifications produced by this Working Group will include security and privacy considerations. Specifically, the user must always be in control of privacy-sensitive information that may be conveyed through the APIs, such as the visibility or access to secondary displays.

Members of the Working Group should review other working groups' deliverables that are identified as being relevant to the Working Group's mission.

2.1 Success Criteria

To advance to Proposed Recommendation, each specification is expected to have two independent implementations of each feature defined in the specification.

To advance to Proposed Recommendation, interoperability between the independent implementations should be demonstrated. Interoperable user agents hosting the same Presentation API web application should be able to render the same content with the same functionality on supported secondary displays that are compatible with the content to render.

2.2 Out of Scope

The specifications defined by this Working Group abstract away the means of connecting and different connection technologies. For example, the following are out of scope:

- Lower level APIs that expose features of different connection technologies
- How second screens are connected to the primary device (e.g. Video Port, HDMI, WiDi, Miracast, AirPlay)
- How the User Agent prepares and sends the screen contents to the second screen

This Working Group will not define or mandate network protocols for sharing content between user agents and secondary displays. For example, the following are out of scope:

- Discovery of wireless secondary displays by the primary user agent
- Establishment of a messaging channel between the two parties, including message addressing, security and authentication
- Negotiation of a media streaming session between devices
- Network transport of media data

To facilitate interoperability among user agents and display devices and encourage adoption of the API, the group may informatively reference existing suites of protocols, either directly in the Presentation API deliverable or in a non-normative companion Note.

Content mirroring, whereby a web application requests display of the content shown in its own browsing context (i.e., page) on a secondary display, is out of scope. If a web application requests display of itself (same URL) on a connected display, a new browsing context will be created with that URL and rendered on the connected display.

This Working Group will not define or mandate any video or audio codecs.

3. Deliverables

3.1 Recommendation-Track Deliverables

The working group will deliver at least the following specification:

Presentation API

An API that allows a web application to request display of web content on a connected display, with a means to communicate with and control the web content from the initiating page and other authorized pages. The initial version of this document will be copied from the Presentation API W3C Community Group Final Report produced by the W3C Second Screen Presentation Community Group. Further modifications (if any) will be decided upon by this Working Group.

The working group may decide to group the API functions in one or more specifications.

3.2 Other Deliverables

Test suite

A comprehensive test suite for all features of a specification is necessary to ensure the specification's robustness, consistency, and implementability, and to promote interoperability between User Agents. Therefore, each specification must have a companion test suite, which should be completed before transition to Candidate Recommendation, and which must be completed with an implementation report before transition to Proposed Recommendation. Additional tests may be added to the test suite at any stage of the Recommendation track, and the maintenance of a implementation report is encouraged.

Use cases and requirements

The Working Group is strongly encouraging the participants to create and maintain a use cases and requirements document for each specification.

Implementation guidelines

To facilitate interoperability among user agents and display devices and encourage adoption of the API, the group may provide informative guidelines for implementors, either directly as informative notes within the Presentation API or in a separate non-normative group Note.

Other non-normative documents may be created for each specification, for example:

- Primers
- Non-normative schemas for language formats
- Non-normative group notes

3.3 Milestones

Note: The group will document significant changes from this initial schedule on the group home page.

First Public Working Draft	Q4 2014
Last Call Working Draft	Q2 2015
Candidate Recommendation	Q4 2015
Proposed Recommendation	Q2 2016
Recommendation	Q2 2016

4. Dependencies and Liaisons

4.1 Dependencies

The initial draft of the Presentation API was prepared by the Second Screen Presentation Community Group. Upon approval of the Working Group, the Community Group will cease its work on the Presentation API specification. It is expected that the Community Group will recharter to work on other related deliverables where it is not clear enough how to proceed for it to be a work item for a Working Group. The Community Group is only one possible source for work under future WG Charters, but can serve to do initial exploration for some future work items.

The specifications produced by this Working Group adhere to the Web's security model defined in the HTML specification published by the HTML Working Group.

Common web technologies that this Working Group could refer to for messaging include Web Messaging and the Web Socket API defined by the Web Applications Working Group.

Even though the scopes of the APIs are different, some of the use cases that the Presentation API aims to address, in particular projection of web content on second screens connected to the local area network, are also in scope of the Network Service Discovery API worked upon by the Device APIs Working Group. This Working Group will liaise with the Device APIs Working Group, in particular to ensure consistency of supported service discovery mechanisms when a user agent implements both APIs.

This Working Group is not aware of any dependencies other Working Groups' specifications have on this Working Group's specifications.

4.2 Liaisons

The Working Group expects to maintain contacts with at least the following groups and Activities within W3C (in alphabetical order) and ask for reviews of deliverables in Last Call, and where appropriate:

Device APIs Working Group

The Device APIs Working Group defines the Network Service Discovery API that addresses some of the use cases that are in scope of the Second Screen Presentation Working Group.

HTML Working Group

The HTML Working Group's deliverables cover the security model implemented in Web Browsers; this security model imposes limitations on what an extended model for Web Applications can achieve.

Privacy Interest Group

The Second Screen Presentation Working Group intends to secure reviews on its deliverables from the Privacy Interest Group to ensure they offer the right level of protection to users.

Second Screen Presentation Community Group

This group developed the initial version of the Presentation API and will likely continue to explore new features.

Web Accessibility Initiative Protocols and Formats Working Group

To ensure the Presentation API supports accessibility requirements, particularly with regard to interoperability with assistive technologies, and inclusion in the deliverable of guidance for implementing the group's deliverables in ways that support accessibility requirements.

Web Applications Working Group

This group defines relevant or potentially relevant specifications including Web IDL, HTML5 Web Messaging and The Web Socket API.

Web and TV Interest Group

This group provides use cases and requirements for second screen scenarios and thus important input on the Presentation API developed by the Second Screen Presentation Working Group.

Web Real-Time Communications Working Group

This group defines relevant or potentially relevant specifications for establishing peer-to-peer communication channels and for extending the Presentation API to support out-of-scope features such as content mirroring.

Web Security Interest Group

The Second Screen Presentation Working Group intends to secure reviews on its deliverables from the Web Security Interest Group to ensure they offer the right level of security.

4.3 External Groups

The Presentation API does not have strong dependencies on any given set of protocols. The following is a

tentative list of external bodies the Working Group should collaborate with to ensure that the Presentation API can be implemented on top of widely deployed attachment methods for connected displays:

DLNA
The Digital Living Network Alliance references home network protocols that secondary displays may support.

IETF
The IETF develops home network protocols that secondary displays may support.

UPnP Forum
The UPnP Forum develops home network protocols that secondary displays may support.

Wi-Fi Alliance
The Wi-Fi Alliance develops home network protocols that secondary displays may support.

5. Participation

To be successful, the Second Screen Presentation Working Group is expected to have 10 or more active participants for its duration, and to have the participation of industry leaders in fields relevant to the specifications it produces.

The Chairs and specification Editors are expected to contribute one half-day per week towards the Working Group. There is no minimum requirement for other Participants. This Working Group will also allocate the necessary resources for building Test Suites for each specification.

The group also welcomes non-Members to contribute technical submissions for consideration, with the agreement from each participant to Royalty-Free licensing of those submissions under the W3C Patent Policy.

6. Communication

Teleconferences will be conducted on an as-needed basis.

This group primarily conducts its work on the public mailing list public-secondscreen@w3.org. Administrative tasks may be conducted in Member-only communications.

Information about the group (deliverables, participants, face-to-face meetings, teleconferences, etc.) is available from the Second Screen Presentation Working Group home page.

7. Decision Policy

As explained in the W3C Process Document (section 3.3), this group will seek to make decisions when there is consensus and with due process. The expectation is that typically, an editor or other participant makes an initial proposal, which is then refined in discussion with members of the group and other reviewers, and consensus emerges with little formal voting being required. However, if a decision is necessary for timely progress, but consensus is not achieved after careful consideration of the range of views presented, the Chairs should put a question out for voting within the group (allowing for remote asynchronous participation -- using, for example, email and/or web-based survey techniques) and record a decision, along with any objections. The matter should then be considered resolved unless and until new information becomes available.

Any resolution taken in a face-to-face meeting or teleconference is to be considered provisional until 10 working days after the publication of the resolution in draft minutes sent to the working groups mailing list. If no objections are raised on the mailing list within that time, the resolution will be considered to have consensus as a resolution of the Working Group.

This charter is written in accordance with Section 3.4, Votes of the W3C Process Document and includes no voting procedures beyond what the Process Document requires.

8. Patent Policy

This Working Group operates under the W3C Patent Policy (5 February 2004 Version). To promote the widest adoption of Web standards, W3C seeks to issue Recommendations that can be implemented, according to this policy, on a Royalty-Free basis.

For more information about disclosure obligations for this group, please see the W3C Patent Policy Implementation.

9. Document License

Documents produced by this group will be licensed under the W3C Document License. In addition, "Code Components" —Web IDL in sections clearly marked as Web IDL; and W3C defined markup (HTML, CSS, etc.) and computer programming language code clearly marked as code examples— will be licensed under the W3C Software License. The group should use the following copyright statement:

For the purpose of this license, Code Components are:

- Web IDL in sections clearly marked as Web IDL; and
- W3C defined markup (HTML, CSS, etc.) and computer programming language code clearly marked as code examples.

10. About this Charter

This charter for the Second Screen Presentation Working Group has been created according to section 6.2 of the Process Document. In the event of a conflict between this document or the provisions of any charter and the W3C Process, the W3C Process shall take precedence.

Annex 2: Multi-Device Timing Community Group Charter

This is a copy of the Charter of the Multi-Device Timing Community Group prepared by Norut and W3C that sets the mission and scope of the Community Group. The Charter has been slightly re-formatted for better readability in this report. The reference charter is available online at:

http://webtiming.github.io/

Abstract

Timing mechanisms allow operations to be executed at the correct time. The Web already has several mechanisms supporting timed operations, including setTimeout and setInterval, as well as controllers for media frameworks and animations. However, the Web lacks support for multi-device timing. A multi-device timing mechanism would allow timed operations across Web pages hosted by different devices. Multi-device timing is particularly important for the broadcasting industry, as it is the key enabler for web-based secondary device offerings. More generally, multi-device timing has wide utility in communication, collaboration and multi-screen presentation. This Community Group aims to define a common, multi-device, timing mechanism and a practical programming model. This will improve the Web as a platform for time-sensitive, multi-device Web applications.

Multi-device Timing

Timing mechanisms allow operations to be executed at the correct time. Timing has many purposes such as correct time ordering, delay, concurrency/synchronization, periodicity, or perhaps rate adjustments in progress or processing.

Of course, the Web already has several mechanisms supporting timed operations. Most famously setTimeout and setInterval allow fairly precise timed execution of JavaScript and is the basis for timed animations. The HTML5MediaElement also counts as timing mechanism, as it provides timed presentation of continuous media and playback controls. Similarly, the HTMLTrackElement offers time alignment of subtitles and chapter information. WebAnimations is a framework for animations with playback control.

However, the Web lacks support for multi-device timing. All the timing mechanisms above are limited in scope to a single Web page. In contrast, a multi-device timing mechanism would allow timed operations across Web pages hosted by different devices.

Goals

The Community Group aims to define a common, multi-device, timing mechanism and a practical programming model for time-sensitive, multi-device Web applications.

Vision

We envision multi-device timing resources, such as clocks, stop-watches, timeouts or controllers, to be explicitly represented as objects of the Web. This means that timing resources are identified by URL's, and that they are hosted by dedicated Web servers or services. This way, multiple devices may share timing resources essentially by connecting to the same URL. Furthermore, devices may assert control over timing resources by interacting with a server-side object. For example, applications will have the ability to request timing resources to slow down or speed up, and request controllers to pause, resume or skip back to start. Effects of timing control will be equally available to all connected devices. Finally, since multi-device timing resources are true objects of the Web, traditional concepts of ownership and access control apply and may be enforced by servers. This way different permissions (i.e., read-only, read-write) may be given for different end-users and/or different application components.

This vision is in line with the classical client-server architecture of the Web. The vision also enables a programming model, where Web applications can define and use application-level timing resources (e.g., clocks, stop-watches, timeouts or controllers) for application-specific purposes. These timing resources then drive the execution of relevant, time-sensitive aspects of that application. In effect, time-sensitive execution can be remote controlled from anywhere in the world, via the server hosting the timing resources. This approach naturally matches the event driven nature of modern Web applications.

Multi-device timing resources will be modeled after existing timing concepts, i.e., the system clock, stop-watches, timeout-mechanisms, and controllers of media and animation frameworks. In short, the CG aims to provide multi-device implementations of these concepts (with the realization that this may call for modifications to existing concepts and API's).

Multi-device timing implies that timing resources may easily be shared between distributed components in a multi-device Web application, or even between different applications all together. Note also that multi-device timing resources support the abstraction of being simultaneous (in real time). Essentially they implement distributed agreement in time. So, if an application-specific media controller is shared between, say a 100.000 devices, and they all evaluate the controller currentTime property (during playback), at exactly the same moment in time, they ideally should get the same media offset. Furthermore, if one device requests the controller to skip ahead by 5 seconds, this ideally affects all connected clients equally and immediately. Implementations of multi-device timing resources must approximate this ideal behaviour as precisely as possible.

Furthermore, precise multi-device timing must be available for any permutation of device, network connection, OS or Web-browser. In short, multi-device timing resources should be available whenever and wherever the Web is available.

Finally, a main goal for a multi-device programming model is to facilitate easy and flexible combination of time-sensitive application components within multi-device applications. In particular, we imagine a common API between Web browsers and providers of multi-device timing resources. A common API would be a great contribution to the interoperability of time-sensitive application components, and would even allow independent systems to share temporal aspects without necessarily sharing any application data. A common API would also work against a development of proprietary timing solutions and technological "islands".

Scope of Work

The CG will:

1. Define a common API for multi-device timing resources, i.e, clocks, stop-watches, timeouts and controllers.
2. Review existing programming concepts for timed operation, i.e., setTimeout, setInterval, HTML5MediaElement, HTMLTrackElement, WebAnimation, and suggest adjustments to simplify integration with multi-device timing resources.
3. Draft a protocol specification for a common interface between Web browsers and providers of multi-device timing resources.
4. Discuss native support for multi-device timing in Web browsers.

The outcome of this will likely be a combination of bug reports, API proposals and protocol specifications.

The CG will borrow concepts, ideas and solutions from the Shared Motion proposal [SMP] (aka. Media State Vectors [MSV]) as a starting point. Shared Motions is a generic concept for multi-device timing on the Web, supporting clocks, stop-watches, timeouts and a wide variety of controllers. Shared Motion adopts the centralised approach where motions are hosted online by specialised motion services. Research on Shared Motion documents timing errors < 10 ms when Shared Motion is used in distributed synchronization of HTML5 video and audio, by regular, non-optimised Web browsers [SYNQ]. Though the limits of scalability has not yet been formally documented, the light-weight nature of Shared Motion indicates that online timing services may be highly scalable.

Out of Scope

- Timing mechanisms based on assumptions of synchronized system clocks (e.g., NTP) are out of scope, as synchronized system clocks is not a realistic assumption in the Web environment. This is especially true on mobile devices.
- Timing mechanisms based on assumptions about ISP, network carrier, Intranet or other local communication are out of scope. Multi-device timing services should work anywhere the Web works, and can not make any assumptions, except that devices are connected to Internet, have an internal clock, and support TCP communication. In addition, it must continue to work in NAT setups without port forwarding etc.

- Timing mechanisms that can not reliably deliver lip-synch precision or better are out of scope. Although many applications do not require lip-synch, the ability to use a single mechanism independent of precision-requirements is valuable.
- Timing mechanisms based on multicast streaming or pulse propagation are out of scope, as precision is affected by variation in network latency.
- Timing mechanisms that just provide a clock (i.e., read-only) are out of scope. A multi-device timing mechanism for Web must support a programming model allowing application code to interact with and control timing aspects within the application.
- Timing mechanisms that do not scale well are out of scope. For example, multi-device timing in broadcasting scenarios might require individual timing of millions of devices.
- Timing mechanisms that only support one-way asymmetric control are out of scope, as interaction and control generally should not be limited to a single device, but be available from multiple devices. If desirable, asymmetric control may instead be achieved by means of application-specific restrictions layered on top of a symmetric mechanism.
- Data formats for media synchronization and timed operation are out of scope. In this CG we are only concerned with multi-device timing resources. The CG advocates a model where timing resources are cleanly separated from data resources. This implies that timing resources can be used in combination with any type of timed data, and remain relevant for any choice of data delivery mechanism. This model also supports separation of concern, as backend services can be specialized for timing only - essential with respect to high precision and scalability. Data formats for timed data are already covered by other groups within W3C as well as other standards bodies.

Importance

A multi-device timing mechanism would significantly improve the Web as a platform for timing sensitive multi-device applications.

The broadcasting industry in particular provides a host of use-cases for multi-device timing. For example, Web coverage of live sport may require a delay to match latency of broadcasting networks. Live Web content also requires time-shifted presentation to match later on-demand consumption. Web-based companion device offerings should be synchronized with the programs they enrich, both in live and on-demand settings. Accessibility features for TV may include the option of foreign language audio tracks delivered by smart phones, synchronized (lip-synch) with a shared screen. Traveling viewers might prefer the “light” option for TV programs. This could be audio plus light-weight timed HTML5, as an alternative to bandwidth consuming HD signals. Interactive, time-sensitive ads on companion devices may enable new forms of engagement and revenue.

More generally, multi-device timing is key to a number of very important functions in multi-device applications. Timed presentation of same content on different devices constitutes collaborative viewing and may also require collaborative control. For presentation tools, there could be solutions based on remote controlled HTML5 as an alternative to video-based screen sharing. Multi-device timing also enables multi-screen solutions. Visualization of complex timed data models may be split across multiple screens, yet navigated in unison along a common timeline. Multi-device timing may also be used to record data input from distributed sources according to a common clock, and then later to reproduce those timing aspects in multi-device playback.

In summary, multi-device timing comes with profound implications for the Web as common platform for communication, collaboration and presentation. It will likely affect all domains of Web-based activity; in particular broadcasting, online media, education, health, music, industry, and government.

Community and Business Group Process

Terms in this charter that conflict with those of the Community and Business Group Process are void.

Work Limited to Charter Scope

The group will not publish Community Group Reports that are Specifications on topics other than those listed under "Community Group Reports that are Specifications" above. See below for how to modify the charter. The CLA applies to these Community Group Reports.

Contribution Mechanics

For these Reports, Community Group participants agree to send contributions to either the group “contrib” list or to the general group list, with subject line starting "“[short-name-for spec]". When meeting discussion

includes contributions, contributors are expected to record those contributions explicitly on the mailing list as described.

Chair Selection

Participants in this group choose their Chair(s) and can replace their Chair(s) at any time using whatever means they prefer. However, if 5 participants — no two from the same organization — call for an election, the group must use the following process to replace any current Chair(s) with a new Chair, consulting the Community Development Lead on election operations (e.g., voting infrastructure and using RFC 2777).

1. Participants announce their candidacies. Participants have 14 days to announce their candidacies, but this period ends as soon as all participants have announced their intentions. If there is only one candidate, that person becomes the Chair. If there are two or more candidates, there is a vote. Otherwise, nothing changes.
2. Participants vote. Participants have 21 days to vote for a single candidate, but this period ends as soon as all participants have voted. The individual who receives the most votes —no two from the same organization— is elected chair. In case of a tie, RFC2777 is used to break the tie. An elected Chair may appoint co-Chairs.

Participants dissatisfied with the outcome of an election may ask the Community Development Lead to intervene. The Community Development Lead, after evaluating the election, may take any action including no action.

Decision Process

This group will seek to make decisions when there is consensus. When the group discusses an issue on the mailing list and there is a call from the group for assessing consensus, after due consideration of different opinions, the Chair should record a decision and any objections. Participants may call for an online vote if they feel the Chair has not accurately determined the consensus of the group or if the Chair refuses to assess consensus. The call for a vote must specify the duration of the vote which must be at least 7 days and should be no more than 14 days. The Chair must start the vote within 7 days of the request. The decision will be based on the majority of the ballots cast. It is the Chair's responsibility to ensure that the decision process is fair, respects the consensus of the CG, and does not unreasonably favor or discriminate against any group participant or their employer.

Transparency

The group will conduct all of its technical work on its public mailing list. Any decisions reached at any meeting are tentative. Any group participant may object to a decision reached at an online meeting within 7 days of publication of the decision on the mail list. That decision must then be confirmed on the mail list by the Decision Process above.

Amendments to this Charter

The group can decide to work on a proposed amended charter, editing the text using the Decision Process described above. The decision on whether to adopt the amended charter is made by conducting a 30-day vote on the proposed new charter. The new charter, if approved, takes effect on either the proposed date in the charter itself, or 7 days after the result of the election is announced, whichever is later. A new charter must receive 2/3 of the votes cast in the approval vote to pass. The group may make simple corrections to the charter such as deliverable dates by the simpler group decision process rather than this charter amendment process. The group will use the amendment process for any substantive changes to the goals, scope, deliverables, decision process or rules for amending the charter.

References

[SMP] Shared Motion Proposal
https://lists.w3.org/Archives/Public/public-web-and-tv/2014Dec/0016.html/
https://lists.w3.org/Archives/Public/public-web-and-tv/2014Dec/0017.html/

[MSV] Ingar M. Arntzen, Njål T. Borch and Christopher P. Needham. 2013. "The Media State Vector: A unifying concept for multi-device media navigation". In Proceedings of the 5th Workshop on Mobile Video (MoVid

'13). ACM, New York, NY, USA, 61-66.

[SYNQ] Njål T. Borch and Ingar M. Arntzen, “Distributed Synchronization of HTML5 Media” (15/2014), http://norut.no/nb/node/3041/