ETSI TR 126 907 V15.0.0 (2018-07)

Universal Mobile Telecommunications System (UMTS);
LTE;
HTML5 for a new presentation layer in 3GPP services
(3GPP TR 26.907 version 15.0.0 Release 15)

Reference
RTR/TSGS-0426907vf00

Keywords
LTE,UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Report (TR) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Contents

Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document is a study of existing scene description solutions for 3GPP services and an evaluation of HTML5 as an alternative and unified solution for providing a presentation layer for 3GPP services.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] W3C Candidate Recommendation 6 August 2013, HTML5: "A vocabulary and associated APIs for HTML and XHTML", http://www.w3.org/TR/html5/.

[2] W3C Candidate Recommendation 6 August2013: "HTML Canvas 2D Context", http://www.w3.org/TR/2dcontext/.

[3] W3C Recommendation 24 October 2013: "Geolocation API Specification", http://www.w3.org/TR/geolocation-API/.

[4] W3C Candidate Recommendation 11 December 2012: "Server-Sent Events", http://www.w3.org/TR/eventsource/.

[5] W3C Candidate Recommendation 20 September 2012: "The WebSocket API", http://www.w3.org/TR/websockets/.

[6] W3C Recommendation 30 July 2013: "Web Storage", http://www.w3.org/TR/webstorage/.

[7] W3C Working Draft 6 December 2012: "XMLHttpRequest Level 1", http://www.w3.org/TR/XMLHttpRequest/.

[8] W3C Candidate Recommendation 01 May 2012: "Web Workers", http://www.w3.org/TR/workers/.

[9] 3GPP TS 26.234: "Transparent end-to-end Packet-switched Streaming Service (PSS); Protocols and codecs".

[10] 3GPP TS 26.346: "Multimedia Broadcast/Multicast Service (MBMS); Protocols and codecs".

[11] 3GPP TS 26.247: "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)".

[12] 3GPP TS 26.114: "IP Multimedia Sybsystem (IMS); Multimedia Telephony Media handling and Interaction (MTSI)".

[13] 3GPP TS 26.140: "Multimedia Messaging Service (MMS); Media formats and codecs".

[14] 3GPP TS 26.246: "Transparent end-to-end Packet-switched Streaming Service (PSS); 3GPP SMIL language profile".

[15] WAP Forum Specification: "XHTML Mobile Profile", October 2001.

[16] 3GPP TS 26.142: "Dynamic and Interactive Multimedia Scenes (DIMS)".

[17] W3C Candidate Recommendation 09 January 2014: "Media Source Extensions", http://www.w3.org/TR/media-source/.

[18] Void

[19] ISO BMFF Byte Stream Format, http://www.w3.org/2013/12/byte-stream-format-registry/isobmff-byte-stream-format.html

[20] W3C Working Draft 18 February 2014: "Encrypted Media Extensions", http://www.w3.org/TR/encrypted-media/

[21] W3C Candidate Recommendation 09 May 2013: "HTML Media Capture", http://www.w3.org/TR/2013/CR-html-media-capture-20130509/.

[22] W3C Recommendation 17 December 2012: "Navigation Timing", http://www.w3.org/TR/navigation-timing/.

[23] Khronos Group, 01 March 2013, WebGL Specification, version 1.0.2, https://www.khronos.org/registry/webgl/specs/1.0/.

[24] W3C Last Call Working Draft, 25 March 2014: "Web Cryptography API", http://www.w3.org/TR/WebCryptoAPI/.

[25] W3C Candidate Recommendation 04 July 2012: "Indexed Database API", http://www.w3.org/TR/IndexedDB/

[26] IETF RFC 5261: "An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [1].

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1] and the following apply.
An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

API	Application Programming Interface
CSS	Cascading Style Sheet
DIMS	Dynamic and Interactive Multimedia Scenes
DOM	Document Object Model
EME	Encrypted Media Extensions
HTML5	Hyper-Text Mark-up Language
ISO BMFF	ISO Base Media File Format
MBMS	Multimedia Broadcast/Multicast Service
MMS	Multimedia Messaging Service
MSE	Media Source Extensions
PSS	Packet-switched Streaming Service
RAP	Random Access Point
RTC	Real-Time Communications
RTP	Real Time Protocol
SMIL	Synchronized Multimedia Integration Language
SSE	Server-Sent Events
SVG	Scalable Vector Graphics
XML	EXtensible Markup Language

4 Introduction

4.1 HTML5

4.1.1 Introduction

HTML5 was specified with the target of creating convergence among a wide range of interoperable browsers, which resulted from proprietary extensions of earlier HTML versions to fill identified gaps. HTML5 also defines and for the first time error handling procedures to deal with document processing errors to help circumvent minor but common syntactical mistakes. As part of the simplicity goal, HTML5 aims at operating in a plugin-free environment by providing APIs to fulfil most of the plugin needs. Finally, HTML5 aims at simplifying the syntax to ease authoring of web applications.

4.1.2 New Tags

HTML5 defines a set of new elements that are discussed in the following list:

- Video: a standard way to embed video content on a web page without using browser plugins. The playback controls are provided through Javascript.
- Audio: a standard way to embed audio content on a web page without using browser plugins.
- Canvas: a resolution-dependent bitmap canvas together with scripts that are used for rendering graphs, game graphics, or other visual images on the fly.
- Svg: HTML5 provides support for inline SVG fragments as well as a fallback mechanism for browsers that do not support SVG.
- Article: specifies independent self-contained content. An article should make sense on its own and should be distributable independently from the rest of the site.
- Header: specifies a header for a document or section. This element should be used as a container for introductory content or set of navigational links.
- Footer: defines a footer for a document or section. A footer element should contain information about its containing element.
- Output: represents the result of a calculation.
- Source: is used to specify multiple media resources for media elements such as video and audio. The source tag allows to specific alternative video/audio files, which the browser may choose from, based on its media type or codec support.
- Time: defines either a time or a date in the Gregorian calendar, optionally with a time and time-zone.
- Input/capture: although the <input> tag is not new for HTML5, the capture attribute is. This allows for media capture and file upload [21].

4.1.3 New HTML5 APIs

4.1.3.1 Introduction

HTML5 can be considered as an umbrella set of specifications, that includes the markup language specification and additionally a wide range of APIs. The following sections briefly introduce the APIs that are currently part of HTML5.

Only specifications that have reached Candidate Recommendation status in the W3C are described. However, several specifications under development related to areas such as local media capture, peer-to-peer real time communications (WebRTC), sensors (proximity, gyroscope, etc.), rich audio processing, and technologies impacting mobile devices should be considered for any 3GPP presentation layer in the near future.

4.1.3.2 Canvas API

The canvas API [2] defines a canvas element and a set of drawing operations. A canvas element creates a rectangular area on the page. Javascript is used to perform the drawing operations.

4.1.3.3 Cross-document messaging API

HTML5 introduces an API [1] to enable safe communication between different parts of the document (e.g. different frames). Additionally, HTML5 introduces the security concept of origin. This is used to enable secure cross-referencing between documents that are located on the same origin, identified through a shared URL path.

4.1.3.4 Geolocation API

The Geolocation API [3] allows the user to share their location with web applications to receive location-aware, customized services. The geolocation is provided by the browser as latitude and longitude coordinates.

The use of the geolocation API for emergency services requires careful considerations to fullfil regulatory requirements.

4.1.3.5 Audio and Video APIs

HTML5 introduces new media elements [1] to replace old media embedding techniques such as through plugins. The new media elements are embedded natively in the browser environment. The elements can be manipulated through a common scriptable API. Additional APIs have been introduced to provide support for different solutions for adaptive HTTP streaming as described by the following two sections.

Any HTML5 Presentation Layer solution is expected to support the use of codecs specified for 3GPP services.

4.1.3.7 Media Source Extensions

The Media Source Extensions (MSE) API [17] was developed to allow for the integration of media streaming into HTML5 documents. MSE relies on Javascript to build media streams and feed them into a user agent built-in media pipeline for playback. MSE was intentionally designed without restrictions on how the acquisition of the media data is performed, resulting in high flexibility of the solution to operate in different environments and applications.

MSE enables simple content splicing to enable use cases such as ad insertion and media representation switching.

MSE currently supports three different byte stream formats as registered in [17]. It is recommended that the ISO BMFF Byte Stream Format as described in [19] has to be supported.

4.1.3.8 Encrypted Media Extensions

Encrypted Media Extensions (EME) [20] aim at leveraging pre-installed DRM platforms in a generic way and using an open Javascript API. EME can be seen as complementary to MSE to enable a fully integrated media consumption solution in the browser.

4.1.3.9 Forms API

The HTML5 Forms API [1] enables more convenient handling of web forms. The new Forms API adds new types of controls and new functions and attributes for simplified form processing.

4.1.3.10 Server-Sent Events (SSE)

The server-sent events [4] allow a web server to send events to the browser. This achieved through the use of a new element, the EventSource. In SSE, the client opens a long lived connection to the indicated source, sends an initial request for the event stream, and keeps receiving newly incoming events as soon as they become available. If the connection times out, the user agent will automatically reconnect.

4.1.3.11 WebSocket API and protocol

The websocket API [5] defines a full-duplex communication channel that operates through a single socket over the web. Websockets have been introduced to remove the need for steady polling. In addition, websockets save significant bandwidth resources by omitting the transmission of the textual HTTP headers in each request and response segment.

4.1.3.12 Web Storage API and Indexed DB API

The web storage API [6] was defined to replace the widely used browser cookies to overcome their limitations. Cookies are limited in size and they still have to be exchanged between browser and server with every single request. The Web Storage API allows the creation of storage objects at the browser side, where the application can have access to them using Javascript.

4.1.3.13 Indexed DB API

The Indexed DB [25] defines APIs for a database of records holding simple values and hierarchical objects. Each record consists of a key and some value. Moreover, the database maintains indexes over records it stores. An application developer directly uses an API to locate records either by their key or by using an index. A query language can be layered on this API. An indexed database can be implemented using a persistent B-tree data structure.

4.1.3.14 Drag and Drop API

The Drag and Drop API [1] introduces a long available feature for native applications to HTML5-based web applications. Web applications have been using scripting and basic mouse events to mimic the drag and drop behaviour. Those efforts resulted in non-unified behaviour depending on the implementation. The Drag and Drop API defines the relevant mouse events and standardizes the browser behaviour during a drag and drop operation.

4.1.3.15 XMLHttpRequest

XMLHttpRequest [7] is the API that is used for AJAX, which enables browsers to send HTTP requests in the background. In its new version, XMLHttpRequest enables cross-origin requests, progress events, and the exchange of binary data.

4.1.3.16 Web Workers

Web workers [8] were defined in HTML5 to enable multithreading for web applications. This new feature comes to address performance issues with executing scripts that also affect the responsiveness of web applications. Web workers are enabled by introducing a new Worker object. Communication between the application and the web worker is performed using messaging.

4.1.3.17 Navigation Timing

Navigation [22] timing refers to a feature in HTML5 where developers can measure the time for page loads or resource fetches, and can adapt their application's behaviour accordingly.

4.1.3.18 WebGL

The Khronos Group has defined the WebGL specification [23]. WebGL is a 3D rendering context for the HTML5 Canvas specification. It can be considered as a web binding for the OpenGL specification for native 3D rendering.

4.1.3.19 WebCrypto

WebCrypto [24] specification describes a JavaScript API for performing basic cryptographic operations in web applications, such as hashing, signature generation and verification, and encryption and decryption. Additionally, it describes an API for applications to generate and/or manage the keying material necessary to perform these operations. Uses for this API range from user or service authentication, document or code signing, and the confidentiality and integrity of communications.

4.2 Current Scene Description Solutions

4.2.1 Introduction

3GPP services do not have a common solution for a presentation layer. This fragmentation in the scene description landscape for 3GPP services complicates content preparation and UE implementation. A common and modern scene description solution is due in Release 12 to make use of the modern multimedia capabilities of UEs.

Table 1 shows the different presentation solutions and in which 3GPP services they are used.

Table 1: Status of Scene Description Solution

	MBMS [10]	PSS [9]	DASH [11] (see note)	MMS [13]	MTSI [12]
SMIL 2.0 3GPP SMIL profile [14]	-	CM	-	CM	-
DIMS Mobile Profile Level 10 [16]	CM	CM	-	-	-
XHTML Mobile Profile [15]	-	-	-	R	-
NOTE: DASH may have inherited a scene description solution from PSS, but this is not stated anywhere and may be incomplete. CM: conditional mandatory R: required -: no indication					

4.2.2 XHTML Mobile Profile

XHTML Mobile Profile was defined by the Open Mobile Alliance as a markup language that is designed for resource-constrained web clients, such as the ones on mobile phones and set-top boxes. The XHTML Mobile Profile is based on the XHTML Basic profile.

XHTML Mobile Profile requires support for the following modules:

- XHTML Structure module: defines the major structural elements for XHTML (e.g. body, head, html, and title).
- XHTML Text module: defines all the basic text container elements, attributes and their content model (e.g. h1, br, and div).
- XHTML Hypertext module: defines the "a" element to define hypertext links to other resources.
- XHTML List module: provides the elements that enable creating lists (e.g. li).
- XHTML Forms module: provides the form-related elements (e.g. form and input).
- XHTML Basic Tables module: defines the table-related elements (e.g. table, tr, and td).
- XHTML Image module: defines the "img" element to enable embedding images in the document.
- XHTML Object module: provides elements for general-purpose object embedding.
- XHTML Metainformation module: defines the "meta" element that describes information within the declarative portion of a document.
- XHTML Link module: defines the "link" element that can be used to link external resources.
- XHTML Base module: defines the "base" element that can be used to define a base URI against which relative URIs in the document will be resolved.
- XHTML Intrinsic module: defines a set of intrinsic events and attributes that pertain to elements that fire those events (e.g. onload or onsubmit).
- XHTML Style Sheet module: defines the "style" element to be used when declaring internal style sheets.
- XHTML Style Attribute module: defines the "style" attribute.

- XHTML Scripting module: defines elements that are used to contain information related to executable scripts (e.g. script element).
- XHTML Presentation module: defines elements and attributes for simple presentation-related markup (e.g. b and hr).
- XHTML Target module: adds the "target" attribute to control the window where the link is to be opened.
- XHTML inputmode Attribute module: defines a new attribute that is used to indicate the input mode to certain editable fields.

XHTML Mobile Profile supports the use of CSS styles to control the presentation. Style information may be associated with a document as external style sheets, internal style sheets, or as internal style information.

XHTML Mobile Profile supports scripting through the Scripting Module defined by XHTMLMod. The Scripting Module defines elements and attributes used to contain information pertaining to executable scripts. The syntax and semantics of these elements and attributes are defined by HTML4. User agents complying to this profile support ECMAScript and Javascript scripts.

The XHTML Mobile Profile defines a profile of the W3C DOM event model. In particular, several events have been excluded in the Mobile Profile. For instance, event capture and event dispatching have been excluded for simplicity.

4.2.3 DIMS

Dynamic and Interactive Multimedia Scenes (DIMS) defines a scene description framework that supports dynamic and interactive scenes. DIMS allows frame-accurate media synchronization. DIMS services are comprised of a scene description and a set of discrete and continuous media. DIMS defines means to store and stream a media scene description, its updates, and the related events. The scene description is based on the SVG Tiny 1.2 format. DIMS adds several extensions on top of SVG Tiny 1.2 that are borrowed from other specifications such as LASeR and SMIL. DIMS defines its own mechanism for supporting scene updates. It defines the "updates" element as part of a DIMS namespace to link the scene update stream to the main scene.

DIMS also defines two new events to signal orientation changes in mobile devices. The signals may be used by the scene to adjust the content according to the orientation. To enable tune-in at random points, the concept of RAPs in scene updates is defined. Scene updates are a set of update commands that tell the client how to modify the current DOM and at which time this modification needs to be done. The commands include Insert, Delete, Replace, and Add command, which apply directly to the DOM. In addition, some state management command are also introduced. These are the Save, Restore, and Clean commands, which are used to control persistent storage and caching of media scenes at the client side. Finally, an Activate and a Deactivate command are defined to temporarily enable or disable elements in the DOM tree.

DIMS supports scripting through the usage of the mobile profile of the ECMAScript scripting language. DIMS events and event handling are inherited from SVG and XML events.

To support streaming of dynamic media scenes, an RTP payload format is defined in DIMS. The RTP payload format enables tune-in at RAPs as well as error resilience tools to recover from lost update fragments.

4.2.4 3GPP SMIL Language Profile

3GPP SMIL is a markup language based on SMIL Basic and SMIL Scalability Framework. In addition to the modules that are required by the SMIL Basic profile, a set of additional modules are required by the 3GPP SMIL profile. The following sets of modules are included in the 3GPP SMIL profile:

- SMIL 2.0 Content Control Modules: BasicContentControl, SkipContentControl and PrefetchControl
- SMIL 2.0 Layout Module: BasicLayout
- SMIL 2.0 Linking Module: BasicLinking, LinkingAttributes
- SMIL 2.0 Media Object Modules: BasicMedia, MediaClipping, MediaParameter, MediaAccessibility and MediaDescription
- SMIL 2.0 Metainformation Module: Metainformation
- SMIL 2.0 Structure Module: Structure

- SMIL 2.0 Timing and Synchronization Modules: BasicInlineTiming, MinMaxTiming, BasicTimeContainers, RepeatTiming and EventTiming
- SMIL 2.0 Transition Effects Module: BasicTransitions

The Content Control Modules defines elements and attributes for content control. The Layout module defines attributes to control the spatial layout. The Linking module defines elements and attributes for providing hyperlinks between documents and document fragments. The Media Object modules include the basic media elements and the control attributes. The Metainformation module defines the "meta" and "metadata" elements that are included as part of the head element. The Structure module defines the top-level structure elements of the SMIL document. The Timing and Synchronization modules define attributes for timing and synchronizing the playback of media elements. Finally, the Transition Effects module defines a set of transition effects for images and media elements.

5 Feature Analysis

5.1 Presentation layer characteristics

While HTML5 may address the potential use cases, we may also want to consider the differences between the various presentation languages.

Table 2: Comparison to Different Presentation Solutions

Features/Language	HTML5	SVG	SMIL
Vector Graphics (native)	NO (but possible via embedding SVG or WebGL)	YES	NO (but possible via embedding SVG; though limited)
3D Graphics	NO (but possible with embedding WebGL)	NO	NO
Animations	TBD	YES	YES
Media Synchronization	YES (but limited compared to SMIL?)	YES (based on SMIL)	YES
Application features (e.g. app cache, storage, etc.)	YES	NO	NO

5.2 Comparison of SVG Animation based on SMIL vs. JavaScript

Animations in SVG can be done via JavaScript or SMIL. In JavaScript, typically the window object methods setTimeout() and setInterval() are used to create animation. The animation author chooses the rate of change in an object relative to the refresh rate of the graphic. If the rate of change is too large relative to the refresh rate, the animation might appear to be jerky. On the other hand SMIL lets the browser SW handle all of these decisions. The SMIL code declares what the author would like to see happen and the implementation details are left to the browser. For example, consider the code below which uses SMIL to animate an ellipse when a user clicks on a button:

```
<!DOCTYPE html>
<html>
<body>

<input id="startButton" type="button"  value="try it" />

<svg>
<ellipse id="Ellipse" cx="150" cy="100" rx="50" ry="50" fill="blue">
 <animate attributeName="rx" begin="startButton.click"
 dur="4s" values="50;150;50"
 repeatCount="indefinite"/>
</ellipse>
</svg>

</body>
</html>
```

The attribute that is modified is the "rx" attribute. After the button is clicked "rx" increases from 50 pixels to 150 pixels and then back to 50 pixels within a duration of 4 seconds. The refresh rate is not stated and is left to the browser. The SMIL code itself is part of the "animate" element, which is a child element of the object being animated. So the SMIL code is closely integrated with the SVG element.

Below, consider the JavaScript code that accomplishes the same thing:

```
<!DOCTYPE html>
<html>
<body>

<input type="button" onclick="ellipse_animate(50,1)" value="try it" />

<svg>
<ellipse id="Ellipse" cx="150" cy="100" rx="50" ry="50" fill="blue"/>
</svg>

<script>
function ellipse_animate(x,step)
{
 document.getElementById("Ellipse").setAttribute("rx",x);
 if (x>150||x<50)   //  Are we at the boundary and we need to reverse?
   step=-step;
 x+=step;
 setTimeout(function() {ellipse_animate(x,step)},20); //display the new ellipse every
20msec
}
</script>

</body>
</html>
```

In this example the JavaScript code is at least slightly more complex and now the animation script is no longer closely integrated with the SVG element being animated. In addition, the animation author specifies how often the graphic should be redrawn so that the animation is smooth, instead of leaving this for the browser implementation. With SMIL, the programmer is freed from have to script timed loops using the setTimeout() or setInterval() methods.

6 HTML5 Profile for 3GPP Services

6.1 Features in the 3GPP Profile

The current scene description solutions provide support for the following features:

Table 3: Scene Description Features

	Spatial	Structure	Temporal, Scripting, and Media
XHTML	Text module Basic tables module Presentation module	Structure module Hypertext module List module Forms module Metainformation module Link module Base module Style sheet module Inputmode attribute module	Intrinsic module Scripting module Target module
DIMS	Basic shapes, Text, Paths Painting	Styling Linking Fonts Metadata	Interactivity and events Media Elements ECMAScript Scene Updates
SMIL	Layout Module	Linking Module Metainformation Structure Module	Content Control Modules Media Object Modules Timing and Synchronization Modules Transition Effects Module

Based on the above classification, we propose the following recommendations on the HTML5 support in the 3GPP services (MBMS, PSS, DASH, and MMS):

- All HTML5 tags
- Audio and video APIs
- Forms API
- XMLHttpRequest
- Canvas API
- Javascript
- CSS modules
- Media Source Extensions

Support for the following APIs is recommended:

- Encrypted Media Extensions
- Server Sent Events (SSE)

Support for the following features is optional:

- WebSockets
- WebVTT
- SVG
- WebGL
- WebCrypto
- Indexed DB

In addition, the support for scene updates as provided by DIMS is discussed in the next section.

6.2 Support for Scene Updates

6.2.1 Introduction

HTML5 provides an API for performing scene updates in the background. XMLHttpRequest (usually named as AJAX) is used in a scripting environment to load resources in the background. The information loaded may then be used to update the scene using Javascript operations on the DOM. The new presentation layer using HTML5 should require support for XMLHttpRequest.

In addition, for eMBMS services the delivery of scene updates through MBMS download delivery needs to be enabled. In eMBMS services, it needs to be possible to provide scene updates over the unidirectional channel as is the case with DIMS over MBMS.

6.2.2 Scene Update Description

The scene updates are in fact DOM updates to the DOM tree of the HTML5 document, which constitutes the scene. The scene updates may be related to the media, related to the spatial layout, or related to the presentation style of particular parts of the document. The scene update solution should support all three sorts of updates.

Scene updates can be described in different ways. A simple solution is to programmatically hard-code all DOM tree updates using scripting, i.e. Javascript (i.e. the scene updates are provided as Javascript instructions and not as an external file). This has several drawbacks such as the inadequate support for dynamic scenes as the updates would need to be known a-priori, the complex scene authoring as it requires programming skills and a customized script for every single program or presentation.

Javascript instruction itself may include HTTP requests that enable updates to the DOM where the instruction format may remain proprietary. The open issue here is that the delivery formats are transparent to the delivery and the delivery through 3GPP user services such as in MBMS and PSS is to be defined.

Another way of describing DOM updates is to use XHTML and deliver the changes to the actual XHTML as deltas, e.g. using XML patch as described in RFC 5261 [26]. This approach handles the HTML5 document as a regular XML document, which results on restrictions on the subset of tools and features that can be supported. This approach would also result in inefficiencies as the target is not to update the HTML5 document itself, but instead to update the DOM that results from parsing the HTML5. The receiver would have to first generate the new HTML5 and then generate the updated DOM. In the worst case, this would even refresh the whole scene as would be the case with updating the HTML5 document itself.

Another approach is to describe the DOM updates in a separate document that can be updated and modified without having to update the actual HTML5 document. The update of the present document would also not require programming skills and can be automated or supported by authoring tools fairly easily.

We recommend that support for dynamic scene updates be provided for the 3GPP services.

6.2.3 Scene Update Distribution

Once the scene update is authored, it has to be made available to the UEs. The distribution of the update may happen in-band or out of band of the media streaming session. One approach for doing in-band delivery is to handle the scene updates as sparse timed media. In that case, the updates are streamed as a component of the media session. This approach has been followed by the DIMS solution.

In band channels may be a Representation that exclusively carries the scene updates as timed metadata items or that are encoded as timed text (such as WebVTT or 3GPP timed text). This solution may have implications on the client, as it requires the client to dig into the media data to extract the scene update and then pass it to the HtML5 application.

Another way is to send it in band as a separate file, as would be the case when using FLUTE in MBMS environment. For the case of unicast, periodic polling using HTTP where the scene update files are made available via regular web servers is appropriate.

7 Conclusion

It is recommended that HTML5 with the features and APIs as provided in clause 6 of the present document to be adopted to replace all XHTML, DIMS, and SMIL scene description solutions. For the new presentation layer, it is also

recommended to refer to the media codecs in the 3GPP services as the recommended set of media codecs that HTML5 media elements and user agents have to support.

It is also recommended that the necessary procedures to identify and deliver scene updates be developed for both unicast and broadcast distribution of dynamics scene updates in a fully compatible way with HTML5.

Annex A:
Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
06-2014	64	SP-140217			Presented to TSG SA#64 for information		1.0.0
09-2014	65	SP-140480			Presented to TSG SA#64 for approval	1.0.0	2.0.0
09-2014	65				Approved at TSG SA#65	2.0.0	12.0.0
12-2015	70				Version for Release 13	12.0.0	13.0.0

Change history							
Date	**Meeting**	**TDoc**	**CR**	**Rev**	**Cat**	**Subject/Comment**	**New version**
2017-03	75					Version for Release 14	14.0.0
2018-06	80					Version for Release 15	15.0.0

History

Document history		
V15.0.0	July 2018	Publication