Consumer Electronics Association
1919 South Eads Street
Arlington, VA
22202 USA
(866) 858-1555 toll free
(703) 907-7600 main
(703) 907-7601 fax
www.CE.org

To: Users of CEA-2014-A Web-based Protocol and Framework for Remote User Interface on UPnP™ Networks and the Internet (Web4CE)

From: CEA Technology & Standards Department

Date: August 28, 2008

Subject: Erratum

Please note that there are several errors in the published version of CEA-2014-A (July 2007). Changed pages with the corrections are provided. We apologize for any inconvenience.

- Page 3, Section 2.1, line 2 – Broken link
- Page 4, Section 2.1, item [20] – Outdated reference
- Page 5, Section 2.3, item "DLNA Documents" – Missing reference acquisition URL
- Page 28, Section 5.2.1, [Req. 5.2.1.a] bullet t) – errors
- Page 33, Section 5.2.2, [Req. 5.2.2.i] – error
- Page 35, Section 5.3, bullet 8 a) – error
- Page 39, Section 5.4, bullet 1 – error
- Page 41, Section 5.4, bullet 7, 3rd sub-bullet – Outdated reference
- Page 41, Section 5.4, bullet 9 – error and added namespace
- Page 44, Section 5.4.2, [Req. 5.4.2.a, bullet 1 a) and b) – error
- Page 50, Section 5.5.1, 1st bullet (i.e. getData), 1st sentence - error
- Page 50, Section 5.5.1, 1st bullet (i.e. sendData), 1st sentence – error
- Page 55, Section 5.6.2, [Req. 5.6.2.c] – MIME-type of message is missing
- Page 60, Section 5.7.1 [Req. 5.7.1.f], bullet 4) – ambiguous text for playstate 0 and 5
- Page 64, Section 5.7.2 [Req. 5.7.2.a], bullet 1) – error
- Page 64, Section 5.7.2, [Req. 5.7.2.d], 1st sentence – error
- Page 84, Annex A, bullets 1 and 3 – Add name space and remove bullet 3
- Page 87, Annex F, lines 6 and 7 - error

Networked Device Interoperability Guidelines – Expanded: October 2006), http://www.dlna.org/members/guidelines/DLNA_Guidelines_-_20070814.zip

[3] IETF RFC 2616, HyperText Transfer Protocol – HTTP 1.1, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[4] IETF RFC 2109, HTTP State Management Mechanism, February 1997, http://www.ietf.org/rfc/rfc2109.txt

[5] XHTML 1.0, The Extensible HyperText Markup Language (Second Edition), W3C Recommendation 26 January 2000, revised 1, August 2002, http://www.w3.org/TR/2002/REC-xhtml1-20020801/

[6] HTML 4.01 Specification, W3C Recommendation 24 December 1999, http://www.w3.org/TR/1999/REC-html401-19991224/

[7] ECMAscript Language Specification (Third Edition), December 1999, ECMA-262.pdf, http://www.ecma-international.org/publications/standards/Ecma-262.htm

[8] ISO/IEC 10646, Universal Multiple-Octed Coded Character Set (UCS) Collections, http://www.evertype.com/standards/iso10646/ucs-collections.html

[9] IETF RFC 2818, HTTP over TLS, May 2000, http://www.ietf.org/rfc/rfc2818.txt

[10] IETF RFC 2246, The TLS Protocol Version 1.0, January 1999, http://www.ietf.org/rfc/rfc2246.txt

[11] REC-DOM-Level-2-20001113 Document Object Model (DOM) Level 2 Core Specification, Version 1.0, W3C Recommendation 13 November 2000; http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113

[12] REC-DOM-Level-2-20001113 Document Object Model (DOM) Level 2 Style Specification, Version 1.0, W3C Recommendation 13 November 2000; http://www.w3.org/TR/2000/REC-DOM-Level-2-Style-20001113

[13] REC-DOM-Level-2-20001113 Document Object Model (DOM) Level 2 Events Specification, Version 1.0, W3C Recommendation 13 November 2000; http://www.w3.org/TR/2000/REC-DOM-Level-2-Events-20001113

[14] REC-DOM-Level-2-20030109 Document Object Model (DOM) Level 2 HTML Specification, Version 1.0, W3C Recommendation 9 January 2003; http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109

[15] CR-css-tv-20030514 CSS TV Profile 1.0, W3C Candidate Recommendation 14 May 2003, http://www.w3.org/TR/2003/CR-css-tv-20030514

[16] Graphics Interchange Format version 89a, (c)1987,1988,1989,1990, CompuServe Incorporated, Columbus, Ohio, http://www.w3.org/Graphics/GIF/spec-gif89a.txt

[17] Portable Network Graphics (PNG) Specification (Second Edition), http://www.w3.org/TR/PNG

[18] Interface KeyEvent, W3C Working Draft, 23 September 1999, http://www.w3.org/TR/1999/WD-DOM-Level-2-19990923/events.html#Events-KeyEvent

[19] RemoteUIServerDevice:1 Device Template Version 1.01,UPnP Standardized DCP, Sept. 2004, http://www.upnp.org/standardizeddcps/documents/RemoteUIServerDevice1.0.pdf

[20] RemoteUIServer:1 Service Template Version 1.01, UPnP Standardized DCP, September 2004, http://www.upnp.org/standardizeddcps/documents/RemoteUIServerService1.0.pdf

[21] RemoteUIClientDevice:1 Device Template Version 1.01,UPnP Standardized DCP, Sept. 2004, http://www.upnp.org/standardizeddcps/documents/RemoteUIClientService1.0.pdf

[22] RemoteUIClient:1 Service Template Version 1.01, UPnP Standardized DCP, September 2004, http://www.upnp.org/standardizeddcps/documents/RemoteUIClientService1.0.pdf

[23] Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004, http://www.w3.org/TR/2004/REC-xml-20040204/

[24] Transport Layer Security Working Group, SSL Protocol Version 3.0, November 18, 1996, http://wp.netscape.com/eng/ssl3/draft302.txt

[25] IETF RFC 1123, Requirements for Internet Hosts -- Application and Support, Oct 1989, http://www.ietf.org/rfc/rfc1123.txt

[26] Persistent Client State: HTTP Cookies, Netscape Corp., http://wp.netscape.com/newsref/std/cookie_spec.html

[27] Cascading Style Sheets Level 2 (CSS2), W3C Recommendation, 12 May 1998, http://www.w3.org/TR/1998/REC-CSS2-19980512/

[28] Cascading Style Sheets, level 2 revision 1 (CSS 2.1), W3C Working Draft, 13 June 2005, http://www.w3.org/TR/2005/WD-CSS21-20050613/

[29] CSS3 Module: Paged Media, W3C Working Draft, 10 October 2006, http://www.w3.org/TR/2006/WD-css3-page-20061010/

[30] CEA-2027-A, A User Interface Specification for Home Networks Using Web-based Protocols, 7 March 2005

[31] JPEG File Interchange Format, Version 1.02, Eric Hamilton, C-Cube Microsystems, September 1, 1992. http://www.w3.org/Graphics/JPEG/jfif3.pdf

[32] IETF RFC 3268, Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS), http://www.ietf.org/rfc/rfc3268.txt

[33] ConnectionManager:1 Service Template Version 1.01, UPnP Standardized DCP, June 2002, http://www.upnp.org/standardizeddcps/documents/ConnectionManager1.0.pdf

[34] AVTransport:2 Service Template Version 1.01, UPnP Standardized DCP, May 2006, http://www.upnp.org/specs/av/UPnP-av-AVTransport-v2-Service-20060531.pdf

[35] The XMLHttpRequest Object, W3C Working Draft, 27 February 2007, http://www.w3.org/TR/2007/WD-XMLHttpRequest-20070227/

2.2 Informative References

The following references provide relevant information for the CEA-2014 standard.

[36] Web-safe Colors, E. Bezarra, University of Texas, http://www.edb.utexas.edu/multimedia/Web-safe%20color.pdf

[37] Advanced Television Systems Committee, DTV Application Software Environment Level 1 (DASE-1) Part 2, A/100-2, 9 March 2003, http://www.atsc.org/standards/a100.html

[38] Universal Remote Console over HTTP Protocol Specification. URC Consortium. www.myurc.com/TR/urc-http-protocol.

[39] IETF RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax, August 1998, http://www.ietf.org/rfc/rfc2396.txt

[40] IETF RFC 2732, Format for Literal IPv6 Addresses in URLs, December 1999, http://www.ietf.org/rfc/rfc2732.txt

[41] Associating Style Sheets with XML documents Version 1.0, W3C Recommendation 29 June 1999, http://www.w3.org/TR/xml-stylesheet/

[42] Avoiding click-to-activate, M. Wilton-Jones, http://www.howtocreate.co.uk/noclicktoactivate.html

2.3 Reference Acquisition

ANSI/CEA Standards

Global Engineering Documents, World Headquarters, 15 Inverness Way East, Englewood, CO USA 80112-5776; Phone 800-854-7179; Fax 303-397-2740; Internet http://global.ihs.com; E-mail global@ihs.com

ECMA Documents

ECMA International, Rue de Rhône 114, CH-1204, Geneva; Phone +41 22 849 6000; Fax +41 22 849 6001

IETF Documents

Internet Engineering Task Force (IETF) Secretariat, c/o Corporation for National Research Initiatives, 1895 Preston White Drive, Suite 100, Reston, VA 20191-5434, USA; Phone 703-620-8990; Fax 703-620-9071; Internet www.ietf.org, IETF RFCs may be downloaded from www.ietf.org/rfc.html, IETF Internet drafts may be downloaded from www.ietf.org/ID.html

World Wide Web Consortium Documents

World Wide Web Consortium (W3C); Internet http://www.w3.org/

DLNA Documents

DLNA Administration, C/O VTM, Attn: Membership Services. 5440 SW Westgate Drive, Suite 217, Portland, OR 97221; USA Phone: 503.297.0426, Fax: 503.297.1090, Internet: http://www.dlna.org/; Email: admin@dlna.org, http://www.dlna.org/industry/certification/guidelines/Order_the_DLNA_Guidelines.pdf

UPnP™ Documents

UPnP™ Forum, http://www.upnp.org/

3 Conventions and Definitions

3.1 Conventions

The following keywords are used to differentiate levels of requirements and optionality, as follows:

SHALL - A keyword that indicates a mandatory requirement. Designers are required to implement all such mandatory requirements to assure interoperability with other products

Remote UI Clients MAY only support a limited range of transparency values, and hence MAY show an approximation of the actual alpha value.

- **"per-pixel"**: this indicates support for controlling the alpha transparency for each individual pixel of the browser area if placed on top of video (using the mechanisms defined in the CSS TV profile (see bullet 7 of [Req. 5.4.a])). This means that the CSS color "transparent" is supported for the main background-color of the <body> of the XHTML page if placed on top of video, and for all elements inside the body. The CSS attribute "opacity" is additionally supported for the body as well as all elements inside the body. Note that Remote UI Clients MAY only support a limited range of transparency values, and hence MAY show an approximation of the actual alpha value.

Note: the <overlay>-element only refers to the special case of overlaying and alpha-blending XHTML-content on top of video. For XHTML content that does not go on top of video, overlaying and alpha blending MUST always be supported as per the CSS TV profile (see bullet 7 of [Req. 5.4.a])

q) **<overlaylocal>** - indicates whether or not the Remote UI Client supports overlays for local video content, i.e. allows XHTML content to be rendered on top of local video (partially obscuring the video, see Section 5.7.2 for more details). If included, the value of this element SHALL be: (none|on-off|global|per-pixel), whereby the same requirements as defined for element <overlay> SHALL apply.

r) **<notificationscripts**> - indicates whether or not the Remote UI Client allows scripts to be executed inside 3rd party notification content (as defined in Section 5.6.3). If included, the value of this element SHALL be: (true|false).

s) **<save-restore>** - indicates whether or not the Remote UI Client supports the save-restore mechanism (as defined in Section 5.8). If included, the value of this element SHALL be: (true|false).

t) **<mime-extensions>** - One or more of these elements MAY be included to specify a space-separated list of additional mime-types that the browser of the Remote UI Client supports, that have not been standardized within CE-HTML (i.e. as specified in Section 5.4) and that cannot be indicated by one of the <audio_profile> or <video_profile>-elements (as defined by [Req. 5.2.1.c]). An example of such an additional mime-type is application/x-shockwave-flash.

- Note: Client support for a mime-type means support through the use of the <object>-element if the mime-type refers to a type of media content, or the <script>-element if the mime-type refers to a scripting language, or the <style>-element if the mime-type refers to a style-sheet language. Other uses/behavior is not specified.
- Attribute **"version"** provides an informal indication of the version of the media content type is supported. The attribute SHOULD have the format "x.y" (with x and y being numbers).
- Note: If a mime-type is not listed, the server cannot assume any ability of the client to download new plug-ins to support such a mime-type.

u) **<ui_profile>** - A container element that SHALL be included for defining UI capabilities. The **"name"**- attribute of this element defines the UI profile name (see [Req. 5.2.1.b]).

v) **<audio_profile>** - OPTIONAL tag that indicates that the Remote UI Client supports the A/V plugin-object for streamed audio content as defined in Section 7.1. This element has the following attributes:

- **"name"** - specifies that the Remote UI Client supports one of the audio profile names identified/standardized by DLNA [2] that corresponds to the MIME-type also applies to the fetching of content through links within the initial start page.

[Req. 5.2.2.e] A Remote UI Server SHALL make sure that any references to other content, within the content returned by [Req. 5.2.2.c], SHALL adhere to the capabilities of the Remote UI Client that requested the original content. This includes references that have been specified through link-elements, (i.e. <a>, <area>, and <iframe>), form submission, re-direction HTTP directives (such as Location and Refresh), and content that is directly included within an CE-HTML page (such as stylesheets, images, etc.).

[Req. 5.2.2.f] A Remote UI Server SHOULD make sure that all content that has been referenced through link-elements (i.e. <a>, <area>, and <iframe>), form submissions and re-direction HTTP directives (such as Location and Refresh) adheres to one of the following MIME-types: application/ce-html+xml, application/x-ce-html+xml, image/jpeg, image/png or image/gif. The behavior of linking to these MIME-types is defined by [Req. 5.2.2.g]. The behavior of linking to other MIME-types is undefined.

[Req. 5.2.2.g] A Remote UI Client SHALL replace the current UI contents for the surrounding frame (i.e. “window”-object) with the referenced content (that corresponds to one of the MIME-types as specified by [Req. 5.2.2.f]), if:

1) A link as specified by [Req. 5.2.2.f] is followed (include links where only the #-portion of the URL is changed)
2) The location of an iframe is changed (e.g. through window.location or by changing the src-attribute through the XML DOM)
3) A form is submitted.

Note: the behavior of linking to other MIME-types as defined by [Req. 5.2.2.f] is undefined.

[Req. 5.2.2.h] The Remote UI Client SHALL provide a means to navigate back to the last UI state before the UI contents were replaced as specified by [Req. 5.2.2.g], for example by pressing a browser “back”-button, provided that the UI contents have not been removed from the cache.

1) In addition, any change that is made to the ‘location’ value of a window-object (as defined in [Req. 5.4.2.a]), SHALL be stored by the UI Client to facilitate backward navigation. See Annex Q for additional information about the browser back button for UI authors.

[Req. 5.2.2.i] A Remote UI Server SHALL return the following HTTP status code if the server does not have matching content as per [Req. 5.2.2.a] and [Req.5.2.2.c]:

- 501 Not Implemented

As per [3], the HTTP response that includes this HTTP status code may include some content provided by the remote UI server to explain the error message. If such response is included, it SHOULD have MIME-type “text/plain”.

[Req. 5.2.2.j] A Remote UI Client MAY connect to the URL for the standard “SD_UIPROF” profile that every Remote UI Server SHALL support as per [Req. 5.2.1.f] , after it received the HTTP error code as defined by [Req. 5.2.2.i].

5.2.3 Browser Area

Client	Mandatory for every RUIC
Server	N/A

5.3 HTTP Headers

Client	Mandatory for every RUIC
Server	Mandatory for every RUIS

[Req. 5.3.a] Every Remote UI Client and Remote UI Server SHALL support HTTP 1.1 [3] for connecting to CE-HTML compliant content (as specified in Section 5.4), and the XML UI Listing that can be fetched from the <uiListURL> as per [Req. 5.1.1.2.c], with the following additional rules:

1) **Accept**: The use of this header is OPTIONAL.
   If this header is used, it SHALL have the case-insensitive value: “application/ce-html+xml, application/x-ce-html+xml, image/jpeg, image/png, image/gif”, concatenated with the MIME-types of supported <audio_profile> and <video_profile>-elements (as per [Req. 5.2.1.a]), and the MIME-types of possible browser extensions that correspond to the ones indicated in the Remote UI Client’s capability profile.
   Note that the information about A/V formats and supported MIME-types as defined by the profile information sent through the “User-Agent” header always takes preference.

2) **Accept-Charset:** SHALL always be included in every HTTP-request.
   SHALL always include the case-insensitive value: “UTF-8”.

3) **Accept-Encoding:** The use of this header is OPTIONAL.
   If this header is used, it SHALL include the “identity” encoding.

4) **Accept-Language:** SHOULD be included in every HTTP-request in order to support internationalization.

5) **Content-Encoding:** The use of this header is OPTIONAL.
   If this header is used, it SHALL always have case-insensitive value “identity”, unless a client/server has explicitly indicated support for other content encodings by using an Accept-Encoding header.

6) **Content-Language:** SHOULD be included if the HTTP-request/response has a message-body, in order to support internationalization.

7) **Content-Length:** SHALL be included if the HTTP-request/response has a message-body for which the content length is known.

8) **Content-Type:** SHALL be included if the HTTP-request/response has a message-body.
   SHALL be one of the MIME-types supported by a Remote UI Client, i.e. one of the MIME-types supported by CE-HTML (as defined in Section 5.4), or one of the MIME-types of supported <audio_profile> and <video_profile>-elements (as per [Req. 5.2.1.a]), or one of the browser extensions that is listed in the Remote UI Client’s capability description (through the <ext>-element as per [Req. 5.2.1.a]), or “text/xml” for the XML UI Listing (as defined in Section 5.1.1.5).
   a) For XHTML content that is CE-HTML compliant (as defined in Section 5.4) the **Content-Type** header SHALL be given the case-insensitive value ‘application/ce-html+xml; charset=”UTF-8”’, optionally appended with width and height name-value pairs, or ‘application/x-ce-html+xml; charset=”UTF-8”’, optionally appended with width

5.4 XHTML Profile (CE-HTML)

Client	Mandatory for every RUIC
Server	Mandatory for every RUIS

[Req. 5.4.a] Every Remote UI Client and Server SHALL support/adhere to the following XHTML profile.

1) **XHTML 1.0 Strict or Transitional** [5], which SHALL adhere to the additional requirements and recommendations as specified in Annex G. The MIME-type that SHALL be used for compliant XHTML content is ‘application/ce-html+xml’ or ‘application/x-ce-html+xml’, appended with charset=”UTF-8”, and optionally appended with name-value pairs for the width and the height, as specified by HTTP-response header “Content-Type” in [Req. 5.3.a].

2) **ECMA-262** [7], ECMAScript Language Specification, including support for the “javascript:” URL scheme. The MIME-type that SHALL be used for compliant ECMAScript content is “text/javascript”.

3) **DOM 2**, Document Object Model Level 2 Specification, including mandatory support for the ECMAScript language binding of:
   a) **DOM level 2 Core** [11], including the extended XML interfaces, i.e. method hasFeature(DOMString feature, DOMString version) of the DOMImplementation interface returns true for features “Core” and “XML”, and version “2.0”.
   b) **DOM level 2 Style** [12], with at least providing support for all style properties as defined by item 5 below via the CSS sub-module, i.e. method hasFeature(DOMString feature, DOMString version) of the DOMImplementation interface returns true for features “StyleSheets”, “CSS” and “CSS2”, and version “2.0”.
   c) **DOM level 2 Events** [13], with additional support for the KeyEvent interface as defined in Section 5.4.1. The DOM2 MouseEvent interfaces are OPTIONAL on key-only devices, i.e. method hasFeature(DOMString feature, DOMString version) of the DOMImplementation interface returns true for features “Events”, “UIEvents”, “MutationEvents”, “HTMLEvents” and “KeyEvents”, and version “2.0”, and returns true for feature “MouseEvents” if the Remote UI Client supports pointer based input (i.e. the <pointer>-element has value *true* in the Remote UI Client capability description as defined in Section 5.2.1).
   d) **DOM level 2 HTML** [14] subset, which includes the following interfaces:
      - HTMLDocument (excl. “applets”-attribute),
      - HTMLElement,
      - HTMLCollection,
      - HTMLLinkElement,
      - HTMLBodyElement,
      - HTMLImageElement,
      - HTMLAnchorElement,
      - HTMLFormElement,
      - HTMLInputElement,
      - HTMLTextAreaElement,
      - HTMLButtonElement,
      - HTMLSelectElement,
      - HTMLOptGroupElement,

7) **CSS TV Profile1.0** [15], Cascading Style Sheets for TV Level 1, with the following **additional** CSS properties:
   - **text-shadow** property of CSS 2.0 [27].
   - **min-width, max-width, min-height, max-height, border-collapse, vertical-align** (with all values being supported), **overflow, position:fixed**, and **table-layout** properties of CSS2.1 [28].
   - **image-orientation** as defined in Section 7 of the CSS3 Paged Media Module [29] for CSS Print Enhanced.
   - **input-format** which defines two properties "alpha-numeric" and "numeric" to restrict the input values of text area, text input and password elements to alpha-numeric (as defined in [Req 5.4.1.i]) or numeric-only (as defined in [Req. 5.4.1.j]. The default value is "alpha-numeric".

   a) These CSS properties SHALL adhere to the additional requirements and recommendations as specified in Annex H.
   b) The MIME-type that SHALL be used for compliant stylesheets is "text/css".

8) a) **GIF** [16], including support for animated GIFs with framerates up to 12 frames per second.
   b) **JPEG** [31], whereby support for progressive JPEG is optional.
   c) **PNG** [17], with **PNG** Color type 6 (true color) supporting 8 bit channels for R, G, B and alpha, and **PNG** compression method 0 (zlib format).
   d) The MIME-types that SHALL be used for compliant image content are:
   - "image/gif" for **GIF** content,
   - "image/jpeg" for **JPEG** content,
   - "image/png" for **PNG** content.

9) Support for **tagged opcodes replacement**, whereby tagged opcodes (i.e. <op>-elements) inside a CE-HTML page SHALL be replaced with actual physical labels (textual or graphical) of keys on a Remote UI Client. The syntax of the <op>-elements is defined as follows:

```
<!ELEMENT op (#PCDATA)>
<!ATTLIST op
from CDATA "ce-html"
code CDATA #REQUIRED
xmlns CDATA #FIXED "urn:ce-org:cea2014:op"
>
```

   whereby:
   a) The attribute "code" SHALL be one of the VK_* codes as defined in Annex F as a string
   b) The value specified for the <op>-element SHOULD be a default label provided by the Remote UI Server.

   The information provided in Annex E MAY be used to create a DTD with which it is possible to validate CE-HTML content that contains <op>-elements. Annex E also provides an example of using the <op>-element.

Note that a Remote UI Client MAY support more than what is stated above, for example to

[Req. 5.4.1.k] A Remote UI Client SHALL generate a change-event and update the value attribute of a form element in the HTML DOM, once the value inside the form element is being changed by a key-press, i.e. at least before the keyup event occurs.

[Req. 5.4.1.l] A Remote UI Client SHALL generate one or more “keydown” events while a key is being pressed until the key is released, at a repetition rate determined by the client, and SHALL generate a “keyup” event as soon as the key is released.

[Req. 5.4.1.m] A Remote UI Client SHALL offer a means to set focus to the following elements in a CE-HTML page by using key-based input: <a>, <area>, all form elements, <iframe>, and <object>-elements of type “video” as defined in Section 5.7.
1) Upon receiving focus, the Remote UI Client SHALL generate DOM 2 “DOMFocusIn” events for <a> and <area>, and both a DOM 2 “focus” and “DOMFocusIn” event for all form elements, for any registered event listeners.
2) For <iframe>-elements and <object>-elements of type “video” the Remote UI Client SHALL call the event listener that has been specified respectively through the onfocus attribute of the “window” object (see Section 5.4.2) and the onfocus attribute of the A/V scripting object (see Section 5.7). The Remote UI Client MAY not generate DOM 2 focus events in those cases.

[Req. 5.4.1.n] A Remote UI Client SHALL offer a means to move the focus away from the following elements in a CE-HTML page by using key-based input: <a>, <area>, all form elements, <iframe>, and <object>-elements of type “video” as defined in Section 5.7.
1) Upon losing focus, the Remote UI Client SHALL generate DOM 2 “DOMFocusOut” events for <a> and <area>, and both a DOM 2 “blur” and “DOMFocusOut” event for all form elements, for any registered event listeners.
2) For <iframe>-elements and <object>-elements of type “video” the Remote UI Client SHALL call the event listener that has been specified respectively through the onblur attribute of the “window” object (see Section 5.4.2) and the onblur attribute of the A/V scripting object (see Section 5.7). The Remote UI Client MAY not generate DOM 2 focus events in those cases.

[Req. 5.4.1.o] A Remote UI Client SHALL support the use of VK_* strings for attribute “accesskey” that is supported by many of the supported XHTML elements.

1) These VK_* strings SHALL correspond to the VK_* keys as specified in the Remote UI Client capability profile to be supported as keys.
2) The Remote UI Client MAY ignore other values that are not indicated in the Remote UI Client capability profile.

5.4.2 Window / UIContentFrame scripting object

Client	Mandatory for every RUIC
Server	N/A

[Req. 5.4.2.a] A Remote UI Client SHALL provide an instance of the “window” scripting object with at least the methods and properties as defined below:

1) Properties:
   a) **readonly String cea2014_protocol** – returns the name of the specification that defines which methods and properties of the “window” object are supported by the Remote UI Client.

For CE-HTML version 1.0, it returns “CE-HTML”.

b) **readonly String cea2014_protocol version** – returns the highest version number of the protocol that is specified by attribute “cea2014_protocol” that the Remote UI Client supports, as a string. For CE-HTML version 1.0, it returns “1.0”.
c) **HTMLDocument document** – read/write property that provides access to the root of the DOM tree of the document that is currently loaded inside the part of the browser area that corresponds to this “window” object, which can also be an iframe defined inside the main browser area (as defined in Section 5.2.3). This property is of type “HTMLDocument” as defined in the ECMAScript binding Section of [14].
d) **readonly window[..] frames** – returns an array containing “window”-objects for all sub-frames that are direct descendants of the current window/frame, i.e. for all iframes elements defined inside the document.
e) **readonly Number innerHeight** – returns the height (in number of pixels) of the current window/frame.
f) **readonly Number innerWidth** – returns the width (in number of pixels) of the current window/frame.
g) **Location location** – read/write property that provides access to the URL of the document that is currently loaded inside the part of the browser area that corresponds to this “window” object (which can also be an iframe defined inside the main browser area), as an object of type Location, as defined below by [Req. 5.4.2.b].
h) **readonly History history** – returns a reference to Javascript’s History object as defined below by [Req. 5.4.2.c].
i) **readonly String id** – returns the identifier of the current frame corresponding to the ID that has been given to it in the XHTML page (i.e. <iframe id=”..”/>). Note: see bullet 6 of [Req 5.4.a].
j) **readonly String name** – returns the name of the current frame corresponding to the name that has been given to it in the XHTML page (i.e. <iframe name=”..”/>). Note: see bullet 6 of [Req 5.4.a].
k) **String onblur** – read-write property that specifies the function to be called when a “blur” event occurs on the window/frame, i.e. when the window/frame that corresponds to this ”window”-object loses focus.
l) **String onfocus** – read-write property that specifies the function to be called when a “focus” event occurs on the window/frame, i.e. when the window/frame that corresponds to this “window”-object gains focus.
m) **String onkeydown** – read-write property that specifies the function to be called when a “keydown” event (as specified in Section 5.4.1) occurs on the window/frame that corresponds to this “window”-object.
n) **String onkeyup** – read-write property that specifies the function to be called when a “keyup” event (as specified in Section 5.4.1) occurs on the window/frame that corresponds to this Window object.
o) **readonly window parent** – read-only property that returns the parent frame that contains the current frame, up to the frame indicated by property “top” (alias “topmost”). The parent of “window.top” (alias “window.topmost”) returns window.top.
p) **readonly window top** – read-only property that returns the top-most “window” object in the hierarchy of “window” objects for the browser area, i.e. the “window” object that corresponds to the browser area that it has made available for the Remote UI Server with the width and height as indicated in the capability profile (as defined in Section 5.2.3).
q) **readonly Number maxHeight** – returns the same value as “height”. This property is only required for compatibility with CEA-2027-A.

number of a host in the network. The domain of this host SHALL be the same as the domain of the server that served the script that calls this method. Only in cases whereby the client is able to trust the server that served the script that calls this method, this restriction does not apply. If opening the TCP/IP connection is successful, the tcpStatus will change to 1. If unsuccessful, an onStatusChange SHALL be triggered with value 0 (even if the tcpStatus already had value 0).

- **getData(): String** – Returns the message, up to and not including the next single byte character with hexadecimal value 0x04 (i.e. Unicode character \u0004, encoded using UTF-8), that was received via the TCP/IP connection that has been opened through method openPersistentConnection(), and for which the onDataArrival-handler was called.

  Note: The Remote UI Client is responsible for buffering incoming events to prevent incoming events from getting lost.
- **sendData(String data): Boolean** – To send the given data, automatically concatenated with a single byte character with hexadecimal value 0x04 (i.e. Unicode character \u0004, encoded using UTF-8) over the TCP/IP connection that has been opened through method openPersistentConnection() to the Server. This method SHALL not block the execution of subsequent methods, and SHALL return “true” if it has accepted the data to be sent, and SHALL return “false” if the method has not successfully sent the previous message yet. If the tcpStatus was 2 or 3 while calling this method, the tcpStatus will be set to 1.
- **close()** - closes the current TCP/IP connection, discards all send and receive queues and sets tcpStatus to 0.

The following state diagram gives an overview of how Notifsocket works:

notification, the Remote UI Client SHALL alert the user to the impending new notification subscription (including information about the highest priority notification type that will be sent by the Remote UI Server), and provide the user with at least two options:

- subscribe to this notification, and
- do not subscribe to this notification.

This does not exclude an option that allows a user to always accept notifications from the same URL.

c) If the Remote UI Client does not subscribe because the user declined, the **subscribeToNotifications** method SHALL return *false*.

[Req. 5.6.2.b] The HTTP GET request performed by the Remote UI Client as part of this polling-based notification approach SHALL adhere to the HTTP headers requirements as defined in Section 5.3. This includes sending the cookies that have been set for the (domain of the) Remote UI Server along with the poll-request.

[Req. 5.6.2.c] The Remote UI Server to which the notification subscription as defined by [Req. 5.6.2.a] applies SHALL respond to a notification polling HTTP GET request, by using an HTTP response with a message body with MIME-type “text/plain” that SHALL either be empty (if there is currently no notification or the previous notification has expired), or SHALL consist of the following items in this order, each terminated by CRLF (as defined in [3]):

1) **notification ID** - a string which SHALL be unique for each non-duplicated notification fetched from a single notification URL; a Remote UI Client SHALL ignore notifications with notification IDs equal to that of the last displayed notification from the same source
2) **event type** - a string which SHALL equal one of the event types listed in Section 5.6.1, without the “upnp:”-prefix; the priority of the event type SHALL not be higher than the priority of the notification subscription associated with this response, as indicated by the value of the *type* parameter of **subscribeToNotifications**
3) **notification URL** - the URL from which the Remote UI Client MAY fetch the notification document, or this URL with “?cancel” appended to it (see [Req. 5.6.3.h])

[Req. 5.6.2.d] A Remote UI Client SHOULD poll for notifications even when the CE-HTML browser is not active. If a new notification is received, this MAY be notified to the user in a vendor defined way, including direct rendering on the display and using a non-intrusive prompt.

[Req. 5.6.2.e] A Remote UI Client SHOULD restrict the total number of active notification subscriptions to about 10.

5.6.3 Notification Content and Window

Client	Mandatory for every RUIC
Server	Mandatory for every RUIS

[Req. 5.6.3.a] A Remote UI Client SHALL ignore polling-based notification messages (as defined in Section 5.6.2) with a notification ID value equal to that of the last received polling-based notification with the same notification URL.

A Remote UI Client SHALL ignore multicast notification messages (as defined in Section apply to other elements, apply to the <object>-element.

1) Style properties that MAY not apply to the <object>-element include the *opacity* attribute of the CSS TV profile.
2) The following CSS-properties SHALL be supported for embedded video <object>-elements (and MAY therefore be changed using the DOM Level 2 Style module):
width, ***height***, ***position***, ***float***, ***top***, ***left***, ***right***, ***bottom***, ***vertical-align**, **padding***, and ***padding-**** properties, ***margin***, and ***margin-**** properties, ***border***, and ***border-**** properties, ***visibility***, and ***display***.

[Req. 5.7.1.e] If the value of the **<overlay>**-element in the capability description of the Remote UI Client is not set to *"none"*, video objects SHALL support CSS-property ***z-index*** (in addition to what is stated in [Req. 5.7.1.d], in both full-screen and windowed mode. Moreover, the Remote UI Client SHALL support the CSS ***opacity*** property and CSS3 **RGBA** color values with a A < 1, for any non-video XHTML element on top of a video object, following the rules as defined by bullet p) of [Req. 5.2.1.a]. If the value of the <overlay>-element in the capability description of the Remote UI Client is equal to *"none",* video objects SHALL always be rendered on top, and their *z-index* value is ignored.

Note: A negative z-index has no 'special status'; if an element *E* has a z-index *z0*, all elements with a z-index < *z0* are rendered 'behind' *E* and all elements with a z-index > *z0* are rendered 'in front of' *E.*

[Req. 5.7.1.f] The following properties and methods SHALL be supported for audio objects and for video objects:
1) String **data** [RW] - media URL. If the value of *data* is changed while media is playing, playback is stopped (resulting in a play state change). The default value is the empty string. If the value of this attribute is changed, the related *data*-attribute inside the DOM tree SHOULD be changed accordingly.
2) Number **playPosition** [R] - the play position in milliseconds of the media referenced by *data* when *data* refers to a single media item. **playPosition** is the duration of the currently playing media item of a playlist if *data* refers to a playlist.
3) Number **playTime** [R] - the total duration in milliseconds of the media referenced by *data* when *data* refers to a single media item. **playTime** is the duration of the currently playing media item of a playlist if *data* refers to a playlist.
4) Number **playState** [R] - indication of the current play state:
0 - *stopped*; the current media pointed to by *data* has stopped playback
1 - *playing*; the current media pointed to by *data* is currently playing
2 - *paused*; the current media pointed to by *data* has been paused
3 - *connecting*; connecting to media server
4 - *buffering*; the media is being buffered before playback
5 - *finished;* the playback of the current media has reached the end of the media
6 - *error*; an error occurred during media playback
5) Number **error** [R] - error details; only significant if the value of *playState* equals 6:
0 - A/V format not supported
1 - cannot connect to server or connection lost
2 - unidentified error
6) Number **speed** [R] - the current relative playback speed (only differs from 0 if the value of *playState* equals 1).
7) Number **nrTracks** [R] – the number of tracks for the media referenced by *data.* If the

5.7.2 Local video content

Client	Recommended for every RUIC (as per [Req. 5.2.1.c])
Server	n/a

[Req. 5.7.2.a] Combining CE-HTML content with a Remote UI Client's local video (i.e. video content that is not directly associated with the remote UI, but rather under the local control of the remote UI client, such as the output of a TV tuner located at the same device as the remote UI client) SHOULD be supported through the use of this special type of video object:

<object type="video/local”></object>

1) If visible, this video object SHALL by default occupy the full-screen display of the Remote UI Client, unless specified differently through CSS (see Section 5.7.3 for more information).
2) At any moment in time only a single instance of a local video object SHALL exist per CE-HTML browser.
3) Support for this object SHALL be indicated through a <mime-extension>-element with value “video/local” inside the capability description.
4) If the Remote UI Client does not have a local video source, such as a TV tuner, the Remote UI Client MAY ignore this element.

Note: a Remote UI Client MAY provide native support for supporting these <object>-elements, and is not required to support downloading of a plug-in to support these <object>-elements.

[Req. 5.7.2.b] The following CSS property SHALL be supported for a local video <object> element (and MAY therefore be changed using the DOM Level 2 Style module): ***visibility****.*

[Req. 5.7.2.c] If the value of the **<overlaylocal>**-element in the capability description of the Remote UI Client is not set to *"none"*, local video objects SHALL support the CSS-property ***z-index*** in addition to what is stated by [Req. 5.7.2.b]. Moreover, the Remote UI Client SHALL support the CSS opacity attribute and CSS3 RGBA color values with a A < 1 for any non-video XHTML element on top of a local video object, following the rules as defined by bullet q) of [Req. 5.2.1.a]. If the value of the <overlay>-element in the capability description of the Remote UI Client is *”none”*, the local video object SHALL be rendered on top; its z-index value is ignored.

[Req. 5.7.2.d] A Remote UI Client SHALL not block scripts to receive key events for the keys indicated in the Remote UI Client capability profile. Note: *Input focus* is the CE-HTML browser input-focus; a Remote UI Client is still allowed to forward or pass key events through the local video player, as long as it does not block the scripts from receiving the keys that are listed in the remote UI client capability profile, so long as the Remote UI Client is in remote UI mode (i.e. the CE-HTML browser is active).

[Req. 5.7.2.e] A Remote UI server should create an instance of the <object>-element for defining a local video-object in a CE-HTML page by using document.write() or the DOM createElement method defined in a separate Javascript file, instead of directly including an <object>-tag in the CE-HTML page. Note: this is needed to prevent an additional activation step which is required in various new browsers, due to the Eolas patent (see [42] for more information).

Annex E: XHTML DTD with Tagged Opcodes (Informative)

In order to support XHTML content with tagged opcodes, one MAY use the following steps to create a DTD to validate such content:

1) Take the XHTML Transitional or Strict DTD [5] as a starting point, and add the following element definition:

```
<!ELEMENT op (#PCDATA)>
<!ATTLIST op
from CDATA “ce-html”
code CDATA #REQUIRED
xmlns CDATA #FIXED "urn:ce-org:cea2014:op"
>
```

2) Add “| op” to the entity-definition of “inline”, “a.content” and “button.content”.

The following is an example of a CE-HTML file that uses tagged opcodes. For the doctype and namespace we basically follow the XHTML-requirements, as indicated in Section 3.1.1 of the XHTML 1.0 spec, which state that the namespace must be "xmlns="http://www.w3.org/1999/xhtml", and the DOCTYPE must have either the public Identifier: PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" or PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN". However, if you want to validate the document, then this should be done using an adapted DTD, as defined above. How you achieve that is up to the client vendor. Of course, the most logical way is to look at the MIME-type, and if it is ‘application/ce-html+xml’ or ‘application/x-ce-html+xml’ then use a local version of the adapted DTD (e.g. ce-html-1.0-transitional.dtd as in the example below) Note: a Remote UI Client is allowed to use its own DTD for validation, thereby overriding any given DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "ce-html-1.0-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Sample opcode replacement file</title>
</head>
<body>
  <p>
    For assistance, press the
    <op code="VK_HELP">Help</op>
    key.

  </p>
</body>
</html>
```

In this example, if the Remote UI Client’s Remote Control Unit has a “help” key (or a key that is typically used on that device for accessing help), the Remote UI Client must replace the tagged text with the Remote Control Unit’s label for the physical key that will generate a key event for the VK_HELP key, before presenting the document.

```
    const Number VK_PLAY_SPEED_DOWN = 419;
    const Number VK_PLAY_SPEED_RESET = 420;
    const Number VK_RECORD_SPEED_NEXT = 421;
    const Number VK_GO_TO_START = 422;
    const Number VK_GO_TO_END = 423;
    const Number VK_PREV = 424;
    const Number VK_NEXT = 425;
    const Number VK_RANDOM_TOGGLE = 426;
    const Number VK_CHANNEL_UP = 427;
    const Number VK_CHANNEL_DOWN = 428;
    const Number VK_STORE_FAVORITE_0 = 429;
    const Number VK_STORE_FAVORITE_1 = 430;
    const Number VK_STORE_FAVORITE_2 = 431;
    const Number VK_STORE_FAVORITE_3 = 432;
    const Number VK_RECALL_FAVORITE_0 = 433;
    const Number VK_RECALL_FAVORITE_1 = 434;
    const Number VK_RECALL_FAVORITE_2 = 435;
    const Number VK_RECALL_FAVORITE_3 = 436;
    const Number VK_CLEAR_FAVORITE_0 = 437;
    const Number VK_CLEAR_FAVORITE_1 = 438;
    const Number VK_CLEAR_FAVORITE_2 = 439;
    const Number VK_CLEAR_FAVORITE_3 = 440;
    const Number VK_SCAN_CHANNELS_TOGGLE = 441;
    const Number VK_PINP_TOGGLE = 442;
    const Number VK_SPLIT_SCREEN_TOGGLE = 443;
    const Number VK_DISPLAY_SWAP = 444;
    const Number VK_SCREEN_MODE_NEXT = 445;
    const Number VK_VIDEO_MODE_NEXT = 446;
    const Number VK_VOLUME_UP = 447;
    const Number VK_VOLUME_DOWN = 448;
    const Number VK_MUTE = 449;
    const Number VK_SURROUND_MODE_NEXT = 450;
    const Number VK_BALANCE_RIGHT = 451;
    const Number VK_BALANCE_LEFT = 452;
    const Number VK_FADER_FRONT = 453;
    const Number VK_FADER_REAR = 454;
    const Number VK_BASS_BOOST_UP = 455;
    const Number VK_BASS_BOOST_DOWN = 456;
    const Number VK_INFO = 457;
    const Number VK_GUIDE = 458;
    const Number VK_TELETEXT = 459;
    const Number VK_SUBTITLE = 460;
    const Number VK_BACK = 461;
    const Number VK_MENU = 462;
```