SMPTE 363.2M-2002
Revision of
SMPTE 363M-2002

SMPTE STANDARD

for Television — Declarative Data Essence — Content Level 1

Table of contents

1 Scope
2 Normative references
3 Introduction
4 Content formats
5 Transport types
Annex A JFIF guidelines for JPEG
Annex B Examples of integrating television with web pages
Annex C Using enhanced television
Annex D Glossary
Annex E Receiver trigger behavior
Annex F Security model
Annex G Bibliography

1 Scope

This standard defines a standard for the authoring of declarative data content intended to be combined primarily with video and/or audio services, and distributed to data-capable television signal receivers. Declarative content is generally nonprocedural, and most commonly in the form of HTML. However, procedural scripting is also defined.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

EIA-746-A-1988, Transport of Internet Uniform Resource Locator (URL) Information Using TEXT-2 (T-2) Service

SMPTE 343M-2002, Television — Declarative Data Essence — Local Identifier (lid:) URI Scheme

SMPTE 366M-2002, Television — Declarative Data Essence — Document Object Model Level 0 (DOM-0) and Related Object Environment

IETF RFC 1952, GZIP File Format Specification Version 4.3

IETF RFC 2046, Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types

Approved
22 August 2002

IETF RFC 2068, Hypertext Transfer Protocol — HTTP/1.1

IETF RFC 2110, MIME E-mail Encapsulation of Aggregate Documents, Such as HTML (MHTML)

IETF RFC 2318, The Text/CSS Media Type

IETF RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax

IETF RFC 2838, Uniform Resource Identifiers for Television Broadcasts

ISO 8859-1-1998, 8-Bit Single-Byte Coded Graphic Character Sets — Part 1: Latin Alphabet No. 1

ISO/IEC 10918-1:1994, Information Technology — Digital Compression and Coding of Continuous-Tone Still Images: Requirements and Guidelines

ISO/IEC 11578:1996, Information Technology — Open Systems Interconnection — Remote Procedure Call (RPC), Annex A, Universal Unique Identifier

ISO/IEC 16262:1998, Information Technology —ECMAScript Language Specification

W3C Recommendation, Cascading Style Sheets, Level 1 (CSS1)

W3C Recommendation, HyperText Markup Language, version 4.0

W3C Recommendation, PNG (Portable Network Graphics) Specification

IANA Media Types, application/tve-trigger, (http://www.isi.edu/in-notes/iana/assignments/media-types/media-types), Registration of MIME media type application/tve-trigger (http://www.isi.edu/in-notes/iana/assignments/media-types/application/tve-trigger)

Object Management Group (OMG) CORBA/IIOP 2.3.1, The Common Object Request Broker: Architecture and Specification, Section 3, OMG IDL Syntax and Semantics

3 Introduction

This document defines a standard for the authoring of declarative data content intended to be combined with video and audio services and distributed eventually to data-capable television receivers.

It is assumed that the reader is familiar with all the normative references; basic concepts are not discussed in this standard.

This specification for enhanced television programming uses existing Internet technologies. It delivers enhanced television programming over both analog and digital video systems using terrestrial, cable, satellite, and Internet networks. The specification can be used in both one-way broadcast and two-way video systems, and is designed to be compatible with all international standards for both analog and digital video systems.

A central design point is to use existing standards wherever possible and to minimize the creation of new specifications. Existing web standards, with only minimal extensions for television integration, provide a rich set of capabilities for building enhanced television content in today's marketplace. This standard references full existing specifications for HTML, ECMAScript, DOM, CSS, and media types as the basis of the content specification. The specification is not a limit on what content can be sent, but rather provides a common set of capabilities so that content developers can author content once and play on the maximum number of receivers.

Another key design goal is to provide a single solution that would work on a wide variety of networks. This design is capable of running on both analog and digital video systems as well as networks with no video at all. The specification also supports transmission across terrestrial (over the air), cable, and satellite systems as well as over the Internet. In addition, it will also bridge between networks — for example, data on an analog terrestrial broadcast must easily bridge to a digital cable system. This design goal was achieved through the definition of a transport-independent content format. This document defines two transports — one for broadcast data and one for data pulled through a return path.

Although this specification has the capability to run on any video network, a complete specification requires a specific binding to each video network standard in order to ensure true interoperability. There are many roles in the production and delivery of television enhancements. This standard refers to three key roles: content creator, transport operator, and receiver. The content creator originates the content components of the enhancement including graphics, layout, interaction, and triggers. The transport operator runs a video delivery infrastructure (terrestrial, cable, satellite, or other) that includes a transport for the data. The receiver is a hardware and software implementation (television, set-top box, or personal computer) that decodes and plays this content. A particular group or company may participate as one, two, or all three of these roles.

4 Content formats

The foundation for the declarative content is existing web standards. Mandatory support is required for the following standard specifications:

– HTML 4.0 ([W3C HyperText Markup Language]; all three document type definitions);

– CSS1 (W3C Cascading Style Sheets, Level 1);

– ECMAScript (ISO/IEC 16262);

– DOM-0 (SMPTE 366M).

4.1 Fonts

The following two fonts are "default" and should be expected by authors to reside in the receiver for Latin character (ISO/IEC 8859-1) markets:

– Tieresias, or another sans-serif font with equivalent metrics;

– Monospace.

The different kinds of these basic font families are defined in table 1.

The CSS keyword font size is defined in table 2.

Table 1 – Basic font families

Family	Size (pixels)	Weight	Style
Tieresias	12	Normal	Normal
Tieresias	24	Normal	Normal
Tieresias	18	Normal	Normal
Tieresias	14	Normal	Normal
Tieresias	10	Normal	Normal
Tieresias	9	Normal	Normal
Tieresias	12	Bold	Normal
Tieresias	24	Bold	Normal
Tieresias	18	Bold	Normal
Tieresias	14	Bold	Normal
Tieresias	10	Bold	Normal
Tieresias	9	Bold	Normal
Tieresias	12	Normal	Italic
Monospace	12	Normal	Normal

Table 2 – CSS font size

Font size name	Height (mm)
xx-small	4
x-small	6
small	8
medium	12
large	18
x-large	25
xx-large	40

4.2 ECMAScript support

Authors shall assume ECMAScript support by receivers compliant with SMPTE 366M. ECMAScript syntax should be both parsed and executed. All native objects and methods should be fully implemented and all language syntax and semantics supported. If authors use script objects, methods, or attributes not defined in DOM-0, such as to target a superset of DOM-0 supported by a specific browser implementation, it is the author's responsibility to insert conditional logic that will only present that script for parsing to the intended browser. Authors are advised of possible implementation limitations as follows:

– Array sizes may be limited to 32,768 elements;
– The number data type may be implemented as a 32-bit floating point value;
– The method, Object.prototype, may not be supported.

4.2.1 Cookie support

Authors may count on 1kB for session cookies. Cookie support is not assumed to be persistent when a receiver is turned off.

4.3 Content type support

Because the architecture supports one-way broadcast of data, content creators cannot customize the content for each receiver as they do today with two-way HTTP. The following is a base profile of supported MIME types that may be included in DDE content:

– text/html (HTML 4.0);
– text/plain;
– text/css (CSS1 only);
– image/png (no progressive encoding);

– image/jpeg (no progressive encoding);
– audio/basic.

4.3.1 Image/png

All image/png content is expected to be cached. All possible syntax shall be permitted in the content. All features are expected to be decoded and rendered except as follows:

– Non-square pixels;
– Gamma correction;
– Chroma correction;
– Progressive display

4.3.2 Image/jpeg

All encoding formats are permitted in the content. However, it is only expected that full JFIF (annex A) guidelines may be decoded and rendered. The thumbnail image may be ignored.

4.4 Triggers

Triggers are real-time events delivered for the enhanced television program.

Note that receiver implementations may set their own policy for allowing users to turn on or off enhanced television content, and can use trigger arrival as a signal to notify users of enhanced content availability. They are also free to decide how to turn on enhancements and how to enable the user to choose among enhancements.

Triggers always include a URL, and may optionally also include a human-readable name, an expiration date, and a script. Triggers that include a *name* attribute may be used to initiate an enhancement either automatically, or with user confirmation. If an enhancement is not currently loaded, the expiration has not been reached, and the trigger contains a *script* attribute, then the script is executed through the trigger receiver object when the page is loaded, and after all OnLoad events have fired for the enhancement. If a replaceable enhancement is currently loaded and a trigger is received whose URL does not match the currently loaded top-level page, the trigger expiration has not been reached, the trigger includes a *name* attribute and a *script* attribute; then the new enhancement can be offered to the user or automatically loaded. If the top-level page of the new enhancement is loaded, the script attribute will be evaluated and executed after loading is completed. If an enhancement is currently loaded and a trigger with a script attribute is received with a URL that matches the current top-level page, and the expiration has not been reached, the script attribute will be immediately evaluated and executed. If duplicate triggers with script attributes are received in sequence, script attributes will be evaluated and executed for each instance. The initial top-level page for that enhancement is indicated by the URL in that trigger. Triggers that do not include a *name* attribute are not intended to initiate an enhancement, but should be processed only as events which affect (through the *script* attribute) enhancements that are currently active. If the URL matches the current top-level page and the expiration has not been reached, the script is executed through the trigger receiver object. When testing for a match, parameters and fragment identifiers (i.e., characters in the URL including and following the first "?" or "#" character) in a URL are ignored.

Triggers are text based, and their syntax follows the basic format of the EIA-746-A standard (7-bit ASCII, the high-order bit of the first byte must be "0").

NOTE – The triggers follow the syntax of EIA-746-A, but may be transported in ways appropriate for the transport, such as multicast IP packets for example, rather than using the EIA/CEA-608-B system.

All triggers are text-based and must begin with ASCII '<'. All other values for the first byte are reserved. These reserved values may be used in the future to signal additional nontext based messages. Receivers may ignore any trigger that does not begin with the '<' in the first byte.

The general format for triggers (consistent with EIA-746-A) is a required URL followed by zero or more attribute/value pairs and an optional checksum:

<url> [$attr_1$:val_1][$attr_2$:val_2]...[$attr_n$:val_n][checksum]

The requirement to provide a checksum is transport binding specific. If required by the transport binding, the checksum must come at the end of the trigger. It is required for transport A and not required for transport B.

– Character set: All characters are based on ISO/IEC 8859-1 character set (also known as Latin-1 and compatible with US-ASCII) in the range 0x20 and 0x7e. Any need for characters outside of this range (or excluded by attribute limits below) must be encoded using the standard Internet URL mechanism of the percent character ("%") followed by the two-digit hexadecimal value of the character in ISO/IEC 8859-1.

– Trigger: The trigger begins with a required URL:

<url>: The URL is enclosed in angle brackets (e.g., <http://xyz.com/fun.html>). Although any URL can be sent in this syntax, content level 1 only requires support for http: and lid: URL schemes.

The following attribute/value pairs are defined:

[**name**:*string*]: The name attribute provides a readable text description (e.g., [name:Find Out More]). The string is any string of characters between 0x20 and 0x7e except square brackets (0x5b and 0x5d) and angle brackets (0x3c and 0x3e). The name attribute can be abbreviated as the single letter "n" (e.g., [n:Find Out More]).

[**expires**:*time*]: The expires attribute provides an expiration date, after which the link is no longer valid (e.g., [expires:19971223]). The time conforms to the ISO 8601 standard, except that it is assumed to be UTC unless the time zone is specified. A recommended usage is the form *yyyymmdd*T*hhmmss*, where the capital letter "T" separates the date from the time. It is possible to shorten the time string by reducing the resolution. For example, *yyyymmdd*T*hhmm* (no seconds specified) is valid, as is simply *yyyymmdd* (no time specified at all). When no time is specified, expiration is at the beginning of the specified day. The expires attribute can be abbreviated as the single letter "e" (e.g., [e:19971223]).

[**script**:*string*]: The script attribute provides a script to execute within the context of the page containing the trigger receiver object (e.g., [script:shownews()]). The string is an ECMAScript statement. The form of statement shall be in accordance with the StatementList nonterminal of ECMAScript. The script attribute can be abbreviated as the single letter "s" (e.g., [s:shownews ()]). An example of a script attribute used to navigate a frame within a page to a new URL:

[script:frame1.location.href="http://atv.com/f1"]

[**tve**:*version*]: The tve attribute indicates to the receiver that the content described in the trigger is conformant to the content specification level, *version*. For example, [tve:1.0]. The tve: attribute can be abbreviated as the single letter "v". The *version* can be abbreviated to a single digit when the *version* ends in ".0" (e.g., [v:1] is the same as [tve:1.0]).

The optional checksum must come at the end of the trigger.

NOTE – EIA-746-A requires the inclusion of a checksum to ensure data integrity over line 21 bindings. In other bindings, such as IP, this may not be necessary, and is not required.

[checksum]: The checksum is provided to detect data corruption. To compute the checksum, adjacent characters in the string (starting with the left angle bracket) are paired to form 16-bit integers; if there is an odd number of characters, the final character is paired with a byte of zeros. The checksum is computed so that the ones complement of all of these 16-bit integers plus the checksum equals the 16-bit integer with all 1 bits (0 in ones

complement arithmetic.) This checksum is identical to that used in the Internet Protocol (described in IETF RFC 791); further details on the computation of this checksum are given in IETF RFC 1071. This 16-bit checksum is transmitted as four hexadecimal digits in square brackets following the right square bracket of the final attribute/value pair (or following the right angle bracket if there are no attribute/value pairs). The checksum is sent in network byte order, with the most significant byte sent first. Because the checksum characters themselves (including the surrounding square brackets) are not included in the calculation of the checksum, they must be stripped from the string before the checksum is recalculated there. Characters outside the range 0x20 to 0x7e (including the second byte of two-byte control codes) shall not be included in the checksum calculation.

Other attributes could be defined at a later date. However, all other single character attribute names are set aside for future definition. Receivers may ignore attributes they do not understand.

Using the description above, all the following are valid trigger strings:

<http://xyz.com/fun.html>

<http://xyz.com/fun.html>[name:Find Out More!]

<lid://xyz.com/fun.html>[n:Find Out More!]

<lid://xyz.com/fun.html>[n:Fun!][e:19991231T115959]
[s:frame1.location="http://atv.com/frame1.htm"]

<http://www.newmfr.com>[name:New] [tve:1] [C015]

NOTE – If a trigger does not contain a [name:] attribute, the enhancement referenced by the trigger should not be presented to the user. Only the last example is a Trigger valid for transport A, because it includes the [tve:] attribute and a checksum, both of which are required for transport A.

4.4.1 More on trigger behavior

This clause provides more description of expected trigger behavior.

Triggers may be repeated. This is commonly done to increase the chance that a Trigger will be received correctly, and to provide the opportunity to initiate an enhancement throughout a program by a viewer who joins the program in progress. However, content authors must take care that re-execution of a repeated Trigger by receivers won't break the presentation. For instance, Trigger scripts explicitly can check for whatever context or state is required before performing their function.

When the tve-trigger.releasable is set to false, and a document is loaded, the following trigger shall be ignored:

<new-URL>attributes

However, the following trigger shall always work, independent of the state of tve-trigger.releasable:

<current-URL>[s:window.top.location.href=<new-URL>]attributes

More information on the expected behavior of triggers in a receiver can be found in annex E.

4.5 Trigger receiver object

Television enhancement HTML pages that expect to have triggers sent to them via a trigger stream shall use the HTML object tag to include one and only one trigger receiver object on a page. The trigger receiver object, implemented by the receiver, processes triggers for the associated enhancement in the context of the page

containing the object. The content type for this object is *"application/tve-trigger"*. If a page consists of multiple frames, only one may contain a receiver object.

Sample instantiation:

```
<OBJECT TYPE="application/tve-trigger" ID="triggerReceiverObj">

</OBJECT>
```

Trigger properties –

enabled: A boolean, indicating if the triggers are enabled. The default value is true (read/write).

sourceId: A string containing the ASCII-hex encoded UUID attribute for the announcement for this stream. SourceId is null if the UUID was not set for the enhancement. The UUID should uniquely identify the enhancement (for example, a different UUID for each program), and can be accessed using the trigger receiver object. In analog television and many types of digital television, broadcast data are tied tightly to A/V. Each virtual channel has its own private network associated with it. In other systems, enhancements for many virtual channels can be carried on the same network. These systems can use the UUID to link a television broadcast with a particular enhancement (read only).

releasable: A boolean indicating that the currently displayed top-level page associated with the active enhancement can be released and may be automatically replaced with a new resource when a valid trigger containing a new URL is received. Such a trigger must contain a [name:] attribute. The default value is false. The currently displayed top-level page shall always be capable of replacing itself with a new resource.

backChannel: A string indicating the availability and state of a backchannel to the Internet on the current receiver. When backChannel returns *permanent* or *connected*, receivers can generally perform HTTP get or post methods and expect realtime responses. When backChannel returns *disconnected*, receivers can also expect to perform HTTP get or post methods, but there will be an indeterminate delay while a connection is established. When backChannel returns *unavailable*, no HTTP get or post methods can be performed. No standard behavior can be assumed when any other value is returned. Value is one of:

- Permanent — Always connected;
- Connected — Currently connected, but not always;
- Disconnected — Not currently connected, but can connect;
- Unavailable — Never connected.

contentLevel: A number that corresponds to the content level of the receiver. For this specification, it is 1.0.

4.5.1 IDL for the application/tve-trigger object

```
Interface Trigger {
   attribute boolean enabled;
   readonly attribute string sourceId;
   attribute boolean releasable;
   readonly attribute string backChannel;
   readonly attribute double contentLevel;
};
```

4.5.2 ECMAScript binding

```
Object Trigger
   Properties:
      Boolean enabled
      String sourceId (readonly);
      Boolean releasable
      String backChannel (readonly)
      Number contentLevel (readonly)
};
```

4.6 URI schemes

4.6.1 Integrating television with web pages

Use the "tv:" URL to reference a broadcast television channel. The "tv:" URL may be used anywhere that a URL may reference an image.

Examples of "tv:" URL usage include the object, img, body, frameset, a, and table tags. For examples with specific HTML syntax, see annex B.

Use of tv: here does not require support for the full syntax of IETF RFC 2838, but only, literally, "tv:".

The tv:URL can also be used for navigation, in order to make window.location.href="tv:" and display a full screen television image. The following example HTML code located in the top page will navigate to full screen:

```
<a href="tv:">Click here for full screen TV</a>
```

When window.location.href=="tv:", content authors should assume that when sending a new enhancement (i.e., an enhancement with a different URL from the one just before navigating to full screen), the new enhancement is either offered to the user, or automatically activated and displayed. In addition, the triggerReceiverObj of the immediately preceding enhancement is destroyed, just as though a navigation had occurred to a page which contained no triggerReceiverObj. This means that trigger scripts for the enhancement loaded prior to navigating to full screen are not processed.

4.6.2 Local identifier URL scheme (lid:)

Content delivered by a one-way broadcast is not necessarily available on demand, as it is when delivered by HTTP or FTP. For such content, it is necessary to have a local name for each resource. To support cross-references within the content (for use in hyperlinks or to embed one piece of content in another), these local names must be location independent.

The "lid:" URL scheme, as defined by SMPTE 343M, enables content creators to assign unique identifiers to each resource relative to a given namespace. Thus, the author can establish a new namespace for a set of content and then use simple, human-readable names for all resources within that space. The "lid:" scheme is used to identify resources that should be stored locally by a broadcast capable receiver platform and are not accessible via the Internet.

4.6.3 The javascript: URI scheme

NOTE – Use of the term "javascript" for the URI scheme name is historical and not meant to suggest that it refers to anything other than ECMAScript in DDE-1.

The special "javascript:" URI scheme may be used in some contexts where a URI is permitted and where the result of accessing the resource associated with the URI may produce content of the media type text/html. Specifically, it may be used in the following contexts:

– A element, HREF attribute
– AREA element, HREF attribute
– FRAME element, SRC attribute
– IFRAME element, SRC attribute

The behavior of "javascript:" in any other contexts is unspecified.

The "javascript:" URI scheme has the following syntax:

javascriptUri : "javascript:" statementList

where the construct statementList shall be nonempty and shall adhere to the syntax prescribed by ISO 16262 (EMCAScript 2nd Ed.), section 12.1. Furthermore, the scheme-specific-part of a javascriptUri (i.e., the statementList) shall satisfy the syntax of the opaque_part nonterminal of the generic URI syntax prescribed by RFC 2396.

The resolution of the "javascript:" URI shall adhere to the following procedure or a logical equivalent thereof:

1 Evaluate statementList in an execution content identical to that used to evaluate Global Code, as further described by ISO 16262, section 10.2.1;

2 If the resulting value produced by (1) is not the undefined value, then convert the resulting value to the String Type; if the resulting string value is nonempty and complete, valid, DDE-1 document of media type text/html, use the resulting string value as the content of the resource accessed by this URI;

3 If the resulting value produced by (1) is the undefined value, take no further action.

NOTES

1 If the resulting string is nonempty and invalid DDE-1 content of type text/html, the result of accessing the resource associated with the URI is unspecified.

2 The form of the "javascript:" URI which does not specify a statement list after the scheme component is not supported by DDE-1; in certain legacy browsers, such a URI, when accessed through an A element, would cause a javascript interpreter user interface to be presented to the end-user to permit the direct entry of javascript statements. This feature is not supported by DDE-1.

3 If any invocation of document.write ( ) within statementList would modify the document which evaluates the javascriptUri (i.e., the document in which "javascript:" statementList is located), then the result of accessing the resource associated with the URI is unspecified.

4.7 Content caching

Content authors should expect support for one megabyte (1 MB) of cached simultaneous content. Content creators who want to reach the maximum number of receivers should manage their content to require a high watermark of simultaneous cached content of 1 MB or less. The specific cache size required for each enhancement must be specified in the announcement.

Tve-size represents the maximum size cache needed to hold content for the current page at any time during the program and also all pages reachable by local links. It is the high watermark during the program, not the total

content delivered during the program. Size is measured as the size when the content is delivered (after decompression for content sent using gzip or other compression techniques).

4.7.1 Cache behavior

The content cache shall generally be expected to operate as a FIFO.

Content authors may optionally specify an HTTP Expires entity-header field for content, that gives the date/time after which the content should be considered stale. Before retrieving content out of the cache, the receiver checks the HTTP expiration time before using the content, and discards it without further use if it has expired.

All broadcast content, including CSS, shall be stored in this cache.

The cache shall be flushed on startup, but not on other conditions, including channel change.

Content that is delivered as a single entity shall not be entered into the cache until all subcomponents are received.

Cache contents may be stored compressed and/or along with their transport information. But the cache size upper bound shall be measured relative to the aggregate uncompressed item sizes stripped of all transport headers.

5 Transport types

The display of enhanced television content consists of two steps: delivery of data resources (e.g., HTML pages) and display of named resources synchronized by triggers. All forms of transport involve data delivery and triggers. The capability of networks for one-way and/or two-way communication drives the definition of two models of transport.

This defines two kinds of transport. Transport A is for delivery of triggers by the forward path and the pulling of data by a (required) return path. Transport B is for delivery of triggers and data by the forward path where the return path is optional.

5.1 Transport type A: Return-path data

Most broadcast media define a way for data service text to be delivered with the video signal. In some systems, this is called closed captioning or text mode service; in other systems, this is called teletext or subtitling. For the purposes of this standard, triggers delivered over such mechanisms will be referred to generically as broadcast data triggers.

Some existing broadcast data services provide a mechanism for trigger delivery, but not resource delivery, due to limited bandwidth. Content creators may encode broadcast data triggers using these mechanisms. Broadcast data streams only contain broadcast data triggers so there is no announcement or broadcast content delivery mechanism. Because there are no announcements, the broadcast data service stream is considered to be implicitly announced as a permanent session.

In addition to the other attributes used in triggers, transport type A triggers must contain an additional attribute, "tve:". This attribute is ignored if present in a trigger in transport B since these values are set in transport type B in the announcement. If the "tve:" attribute is not present in a transport type A trigger, the content described in the trigger is not considered to be DDE content.

Television transport operators should use the standard mechanisms for broadcast data trigger transmission for the appropriate medium (EIA, ATSC, DVB, etc.). It is assumed that when the user tunes to a television channel, the receiver locates and delivers broadcast data triggers associated with the television broadcast. Tuning and

decoding broadcast data triggers is implementation and delivery standard specific and is specified in the appropriate binding. A mechanism must be defined for encoding broadcast data triggers for each delivery standard. Because there is no content delivery system, broadcast data triggers usually require two-way Internet connections to fetch content over HTTP, version 1.1.

NOTE – Television transport operators and content creators need to plan to handle the scalability issues associated with large numbers of HTTP requests responding at roughly the same time to broadcast triggers.

5.2 Transport type B: Broadcast data

Transport type B is for true broadcast of both the resource data and triggers. As such, transport type B can run on television broadcast networks without Internet connections, unlike transport type A. An additional Internet connection allowing a return path can be added to provide two-way capabilities like e-commerce or general web browsing.

Transport type B uses announcements to offer one or more enhancements of a television channel. An announcement specifies the location of both the resource stream (the files that provide content) and the trigger stream for an enhancement. Multiple enhancements can be offered as choices that differ on characteristics like language, required cache size, or bandwidth. In addition to locating the files and trigger streams, announcements must be able to provide the following information: language, start and stop times, bandwidth, peak storage size needed for incoming resources, the content level the resources represent, an optional UUID that identifies the content, and an optional string that identifies the broadcast channel for systems that send ATVEF content separately from the audio/video television broadcast. The receiver must be able to start receiving data from only the description broadcast in the announcement.

Transport type B also requires a protocol that provides for delivery of resources. In one-way broadcast systems, this is a one-way resource transfer protocol that allows for broadcast delivery of resources. The resource delivered, no matter what the resource transfer method, must include HTTP headers to package the file on the resource transfer protocol. All resources delivered using resource transfer are named using URLs. These resources are then stored locally, and retrieved from this local storage when referenced using this same URL. Content authors should expect support for local storage and retrieval of content using the "lid:" URL scheme and the familiar "http:" URL scheme. When "lid:" is used, the resources are delivered only through broadcast and are not available on demand. When "http:" is used, the resources that are delivered through broadcast also exist on the World Wide Web and can be requested from the appropriate server using standard HTTP. Sending "http:" resources using resource transfer effectively preloads the local cache, thus avoiding large numbers of simultaneous hits on web servers when those same resources are requested by many receivers. Furthermore, this mechanism allows receivers to view the same content that appears on the web even when no Internet connection is available. Content creators can freely mix resources that use either the "lid:" or "http:" schemes in the same enhanced broadcast. Because the underlying resource transfer protocol is not limited to carrying resources named by any particular URL scheme, some receivers will store and retrieve content named using other URL schemes, such as "ftp:", as well as the required "lid:" and "http:".

Transport type B uses the same syntax for triggers as type A.

5.3 Simultaneous support of transports A and B

A single video program may contain both transport type B and transport type A (e.g., broadcast data triggers) simultaneously. This is advantageous in order to target both IP-based receivers as well as receivers that can only receive broadcast data triggers.

Receivers may choose to support only transport B- based trigger streams and ignore broadcast data triggers (transport A), or receivers may support broadcast data triggers in the absence of transport B-based triggers, or receivers may support broadcast data triggers and transport B-based triggers simultaneously. For receivers that

provide simultaneous support, this specifies the following behavior, which is identical to the treatment of transport B-based triggers on an active stream.

When a broadcast data trigger is encountered, its URL is compared to the URL of the current page. If the URLs match and the trigger contains a script, the script should be executed. If the URLs match but there is no script, the trigger is considered a retransmission of the current page and should be ignored. If the URLs do not match and the trigger contains a name, the trigger is considered a new enhancement and may be offered to the viewer if triggerReceiverObj.releasable is true. If the URLs do not match and there is no name, the trigger should be ignored.

Annex A (normative)
JFIF guidelines for JPEG

This annex further restricts the JPEG encoding and defines a new APP0 field defining a profile commonly known as JFIF. The syntax of a JFIF-profiled JPEG file conforms to the syntax for interchange format defined in annex B of ISO/IEC 10918-1.

The encoding restrictions are:

– Baseline compression only;

– The only color space is YC_bC_r as defined by ITU-R BT.601-5 (normalized to 256 levels). The resulting RGB components shall not be gamma corrected. If only one component is used, that component shall be Y;

– The image orientation is always top down;

– The position of the pixels in subsampled components are defined with respect to the highest resolution component.

The APP0 marker is mandatory right after the SOI marker, and is defined in table A.1.

Table A.1 – APP0 marker

Field	Size (bytes)	Value
APP0	1	JPEG APP0 code
Length	2	Length of the structure including this field
Identifier	5	"JFIF" with null termination and zero parity
Version	2	0x0102
Units	1	The units for (Xdensity, Ydensity) 0x00 = aspect ratio 0x01 = dots per inch 0x02 = dots per centimeter
Xdensity	2	Horizontal pixel density
Ydensity	2	Vertical pixel density
Xthumbnail	1	Horizontal thumbnail pixel density
Ythumbnail	1	Vertical thumbnail pixel density
RGB	3n	Thumbnail RGB values packed in 24 bits, where n = Xthumbnail*Ythumbnail

Annex B (informative)
Examples of integrating television with web pages

The following examples describe how to achieve common design goals for integrating television and web pages. This list is meant to be illustrative rather than exhaustive. The "tv:" URL may be used anywhere that an image URL is also appropriate.

Examples are presented in both HTML 3.2 and HTML 4.0 since the HTML 4.0 specification recommends that tools supporting HTML 4.0 continue to support HTML 3.2.

B.1 How to place television in a web page (using <OBJECT> and <IMG> tags)

The OBJECT and IMG tags are used to place the television picture in a web page, for example:

```
<object data="tv:"   width="60%"   height="60%">
<img src="tv:"   width=320   height=240>
```

B.2 How to place television in a web page that uses tables (using <TABLE> tags)

The TD tag can be used to place the television picture as the background of a table cell, for example:

```
<td width=320   height=240   style="background: url(tv:)">
     Here is content that is overlaid on top of the television
     picture inside this table cell.
</td>
```

B.3 How to overlay a web page over a television background (using <BODY> tag)

The BODY tag is used to specify television as a full-screen background of the web page, for example:

```
HTML 3.2 syntax: <body background="tv:">
HTML 4.0 syntax: <body style="background: url (tv:)">
```

B.4 How to overlay a frame-based web page over a television background (using <FRAMESET> tag)

Many web pages will be frame-based rather than body-tag based. This will allow the program to change the displayed web page while maintaining the same URL for a series of triggers. Since an HTML document that contains a FRAMESET tag cannot contain a BODY tag, it is necessary to specify "tv:" on a FRAMESET when full-screen television is desired beneath the frames, for example:

```
<frameset style="background: url (tv:)" cols="200,*">
```

Each frame in the frameset that wants the full-screen television to show through must specify a transparent background color in the BODY tag of the frame's HTML document, for example:

```
HTML 3.2 syntax: <body bgcolor="transparent">
HTML 4.0 syntax: <body style="background: transparent">
```

B.5 How to transition from a web page back to full-screen television (using <A> tag)

Finally, the use of "tv:" (see IETF RFC 2838) as the href of an anchor tag allows for hyperlinking to full-screen television, for example:

```
<a href="tv:">Click here to go to full-screen TV</a>
```

Annex C (informative)
Using enhanced television

Television enhancements delivered using transport A only contain broadcast data triggers so there is no announcement or broadcast content delivery mechanism. Because there are no announcements, the broadcast data service stream is considered to be implicitly announced as a permanent session.

Television enhancements delivered using transport B are comprised of three related data sources: announcements, content, and triggers.

Announcements have a time period for which they are valid, and also contain information that the client can use optionally to help decide whether to automatically start receiving trigger and content information. This may include type, language, and keyword attributes that provide additional information to the client about the announced enhancements.

When the client sees a new enhancement, it knows that there will be data available on the given content and trigger addresses. The client may present the user with a choice to start receiving trigger and content information, or may do so automatically. The client implementation specifies what kind of user interface, if any, to present. After this confirmation (or automatic behavior), the client receives content and triggers, caching the content and parsing the triggers.

When the client first receives a trigger (containing a URL pointing to some enhancement content), the client may notify the user that the content is available or, alternatively, navigate to that content automatically. Clients may choose not to notify the user if they believe that they cannot display the enhancement, generally because the content referred to by the specified URL is not available.

When an enhancement has either been confirmed by the user, or has been started automatically, the enhancement is displayed. Only one enhancement may be displayed at a time. When new triggers associated with the current enhancement arrive, they are executed or ignored depending on several conditions. If the URL of the trigger matches the URL of the current page and the trigger has a script attribute, the script is played; if there is no script, the trigger is ignored. If the URLs do not match and the trigger has a name attribute, the trigger is considered a new enhancement and is played, offered to the viewer, or ignored depending on other factors described below; if no name attribute, the trigger is ignored.

If a new enhancement is announced while an existing enhancement is being displayed, the client may present the user with the option to begin receiving that announcement data (content and triggers) or do so automatically. Multiple enhancements may be received simultaneously, although only one may be displayed at a time.

When the new enhancement is being received at the same time as an existing enhancement is being displayed, and the new enhancement delivers its first trigger, the client may have one of three behaviors:

– The client ignores the new enhancement trigger until the existing enhancement has been completed;

– It presents the user with the opportunity to navigate to the new enhancement;

– The client automatically navigates to the new enhancement.

It may be important for some triggers to be able to send scripts to the current enhancement without presenting the user with the opportunity to navigate to that enhancement. In this case, no [name:] attribute should be included. This allows enhancements to enforce that the user view them from the beginning and not join in later when a subsequent trigger containing a script is received. If no [name:] attribute is found in the trigger, the user should not be presented with the opportunity to view the enhancement or automatically navigate there. The enhancement's data stream can be used to preload data by sending data before the first trigger that is sent with a [name:] attribute.

Content creators are encouraged to "shut down" their enhancements at the end of the related video content. This means that enhancements should navigate themselves (via trigger scripts or some other scripting mechanism) to full screen television ("tv:") when the program or commercial ends. This will prevent content creators from displaying their enhancement over some unrelated broadcasts and reduce the likelihood of conflicts between producers. Content creators may wish to collaborate with the producers of subsequent programs or commercials to build a single enhancement that spans multiple video segments and may provide some enhanced user experience. For example, a broadcaster may provide a standard top level page that loads and displays all channel content, program content, and advertisement content in separte frames, which can be loaded and displayed at any time in response to script triggers or user initiated scripts executed in the top level page. This permits ad content to be preloaded in hidden frames while other enhancements are displayed, and the quick transition from program enhancements to ad enhancements and back again.

When the time period specified by the announcement is over, clients may automatically end the enhancement to prevent the user from viewing the enhancement over potentially unrelated video.

A property, named triggerReceiverObj.releasable, may be set on the trigger receiver object associated with the current enhancement. When set to *true*, the current enhancement associated with this trigger stream may be automatically replaced with a new enhancement if the client user interface permits this. A subsequent enhancement can become active by sending a trigger which includes a [name:] attribute when the current page's trigger receiver object's triggerReceiverObj.releasable property is *true*.

An example showing a more complex use of triggers and pages follows.

This content would consist of two original source files, an HTML document and a PNG image. The experience consists of a screen with a 60% sized embedded live television object, with some text below it. During the show, a trigger may arrive that will cause an image of the word "MURDER" to appear below the text. If the user chooses to click on the television object, he will be returned to full-screen video, and away from the enhanced experience.

The first would be referred to by the URL:

<lid://nicebroadcaster.com/show27/launch.html>

and consists of the following text:

```
<HTML>
<OBJECT TYPE="application/tve-trigger" ID="triggerReceiverObj">
</OBJECT>

<HEAD>
<TITLE>Day & Night & Day:  The Interactive Experiene</TITLE>
</HEAD>

<SCRIPT LANGUAGE="JavaScript">
function scenechange (imagename)
{
document.sceneimage.src = imagename + ".png";
}
</SCRIPT>

<BODY bgcolor="magenta">

<A href="tv:">
<OBJECTdata="tv:"  width="60%"  height="60%"  align="center">
</OBJECT>
</A>

<BR>
<P>Welcome to the Day & Night & Day Interactive Experience</P><BR>
<Watch below for more information about the current scene!</P><BR>

<IMG name="sceneimage""  align="center"  src="">
</BODY>
</HTML>
```

The second file consists of a PNG image, containing an image of the word "MURDER" in big red letters. Its URL will be specified as <lid://nicebroadcaster.com/show27/murder.png>.

The following trigger would be sent after the data were first transmitted to trigger the beginning of the enhanced television experience:

```
<lid://nicebroadcaster.com/show27/launch.html>[name:Day
  & Night & Day Again Interactive]
```

This trigger packet would also be transmitted periodically later on to allow viewers who tune in late to join in the fun.

Later on, during the program, the content creator might send the following trigger to make the content change to reflect the fact that a murder scene has just begun in the program:

```
<lid://nicebroadcaster.com/show27/launch.html>
  [script:scenechange("murder")]
```

This trigger would cause the active enhancement page (if it matched the URL in the trigger) to execute the ECMAScript function 'scenechange("murder")', which would cause the murder.png image to be displayed within the page. If the specified URL was not currently being displayed, the trigger would be ignored because this trigger does not include a [name:] attribute.

Near the end of their program, they might send the following trigger to tell their interactive application to shut down. This would allow them to more accurately synchronize with the end of the program, rather than relying on the session timing information in the announcement.

```
<lid://nicebroadcaster.com/show27/launch.html>
  [script:window.location="tv:"]
```

Annex D (informative)
Glossary

D.1 announcements: Announcements are used to announce currently available programming to the receiver.

D.2 backchannel: A connection from a receiver to the Internet or back to some server.

D.3 client: The receiver environment that renders the content.

D.4 content creator: In the context of this standard, a content creator has the role of originating the content components of the television enhancement including graphics, layout, interaction, and triggers.

D.5 CSS1 (cascading style sheets, level 1): CSS1 is a simple style sheet mechanism that allows content creators and readers to attach style (e.g., fonts, colors, and spacing) to HTML documents. The CSS1 language is human readable and writable, and expresses style in common desktop publishing terminology.

D.6 DOM (document object model): The document object model is a platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure, and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page.

D.7 ECMAScript: A general purpose, cross-platform programming language.

D.8 enhancement: The content added to a video/audio service.

D.9 forward path: The television broadcast stream.

D.10 HTML (hypertext markup language): A collection of tags typically used in the development of web pages.

D.11 HTTP (hypertext transfer protocol): A set of instructions for communication between a server and a World Wide Web client.

D.12 IETF (Internet Engineering Task Force): The IETF is a large, open community of network designers, operators, vendors, and researchers whose purpose is to coordinate the operation, management, and evolution of the Internet and to resolve short-range and midrange protocol and architectural issues. It is a major source of proposals for protocol standards which are submitted to the IAB for final approval. The IETF meets three times a year and extensive minutes are included in the IETF Proceedings.

D.13 ISO (International Organization for Standardization): A voluntary, nontreaty organization founded in 1946 which is responsible for creating international standards in many areas, including computers and communications. Its members are the national standards organizations of the 89 member countries, including ANSI for the U.S.

D.14 MIME (multipurpose Internet mail extensions): A protocol for allowing e-mail messages to contain various types of media (text, audio, video, images, etc.).

D.15 NABTS: North American Basic Teletext Specification.

D.16 receiver: In the context of this standard, a receiver is a hardware and software implementation (television, set-top box, or personal computer) that decodes and presents declarative content.

D.17 return path: See backchannel.

D.18 triggers: Used to identify the URL and some human-readable string to use in the announcement to the user. In order to announce the availability of the interactive television experience to the user (as opposed to announcing it to the client downloader mechanism).

D.19 television enhancement: A collection of web content displayed in conjunction with a television broadcast as an enhanced or interactive program.

D.20 UUID (universally unique identifier): Also known as GUID (globally unique identifier) is an identifier that is unique across both space and time, with respect to the space of all UUIDs.

D.21 W3C (World Wide Web Consortium): The W3C, an international industry consortium, was founded in October 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability.

Annex E (informative)
Receiver trigger behavior

The behavior associated with the receipt of a trigger depends on a combination of factors relating to the content of the trigger and the state of the currently loaded enhancement (if any). The behavior is described in tables E.1 and E.2 for the following cases:

– Table E.1 — triggers received when no enhancement is currently loaded.

– Table E.2 — triggers received when an enhancement is currently loaded.

For each of these tables, the following preconditions apply:

– If the transport (i.e., transport A or B) and/or binding requires the use of a checksum, then the tables assume that the checksum is valid. If the checksum is present and is not valid, then the trigger is ignored.

– If the transport and/or binding requires that an announcement is received and processed before a trigger associated with that announcement is accepted, then the tables assume that such an announcement has been received and processed. For such transports and/or bindings, if the announcement for which the trigger is associated has not been received and processed, then the trigger is ignored.

– If the trigger contains an expiration date, then the tables assume that the expiration date has not yet been reached. If the expiration date has already been reached, then the trigger is ignored.

E.1 No enhancement currently loaded

If no enhancement is currently loaded, table E.1 describes receiver behavior as a function of the contents of a received trigger.

E.2 An enhancement is currently loaded

If an enhancement is currently loaded, the following conditions describe receiver behavior as a function of the contents of a received trigger:

If the triggerReceiverObj.releasable property of the current tve-trigger receiver object is set to false:

– All triggers of the form <new-URL>attributes shall be ignored.

– All triggers of the form <current-URL>[s:window.top.location.href=<new-URL>]attributes shall be processed in accordance with table E.2. The current enhancement shall always be capable of navigating to a new URL regardless of the state of tve-trigger.releasable.

If the tve-trigger property triggerReceiverObj.releasable of the currently loaded enhancement is set to true, then all triggers are processed in accordance with table E.2.

Table E.1 – No enhancement currently loaded

Trigger contains name	Trigger contains script	Receiver action
No	No	Ignore trigger.
No	Yes	Ignore trigger.
Yes	No	Process enhancement (see note 1).
Yes	Yes	Process enhancement and execute script. (see notes 1 and 2).

NOTES

1 Receiver action with regard to the enhancement is implementation specific. It is expected that the receiver would have one of the following behaviors:

– The enhancement is offered to the user; i.e., the user is presented with the opportunity to activate the enhancement. If the user activates the enhancement, the page is displayed.

– The enhancement is automatically activated and displayed.

2 Note 1 is applied and when the enhancement is activated, the script shall be executed in the context of the document.

Table E.2 – Enhancement currently loaded

Trigger URL= current page URL	Trigger contains name	Trigger contains script	Receiver action
No	No	No	Ignore trigger.
No	No	Yes	Ignore trigger.
No	Yes	No	Process new enhancement (see note 1).
No	Yes	Yes	Process new enhancement and execute script (see notes 1 and 2).
Yes	No	No	Ignore trigger.
Yes	No	Yes	Execute script in the context of the current document.
Yes	Yes	No	Ignore (retransmission of current page).
Yes	Yes	Yes	Execute script in the context of the current document.

NOTES

1 Receiver action with regard to the new enhancement is implementation specific. It is expected that the receiver would have one of the following behaviors:

– The new enhancement is ignored until the existing enhancement has been completed; i.e., the trigger is queued.

– The new enhancement is offered to the user; i.e., the user is presented with the opportunity to activate the new enhancement. If the user activates the new enhancement, the new page is displayed. Only one enhancement shall be active at a time.

– The new enhancement is automatically activated and displayed. Only one enhancement shall be active at a time.

2 Note 1 is applied and when the new enhancement is activated, the script shall be executed in the context of the new document.

Annex F (informative)
Security model

This system assumes that all insertion points of the content described in this standard are trusted sources (or "hosts" in Internet thinking). Therefore, the use of scripts in triggers, and ECMAScript in general, is not presumed to be a security problem for malicious content or viruses any more than the notion of rogue video and audio being inserted into a broadcaster's emission. As with video and audio, the broadcaster is responsible for the quality of the data content of its programming.

The security ramifications of an application of this specification where the insertion points are not all trusted has not been studied and is unknown.

Annex G (informative)
Bibliography

EIA/CEA 608-B-2000, Line 21 Data Services

IETF RFC 791, Internet Protocol — DARPA Internet Program Protocol Specification (IP)

IETF RFC 1071, Computing the Internet Checksum

ISO 8601:2000, Data Elements and Interchange Formats — Information Interchange — Representation of Dates and Times

ITU-R BT.601-5 (10/95), Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide-Screen 16:9 Aspect Ratios