ISMA™
Internet Streaming Media Alliance

Presidio of San Francisco
P.O. Box 29920
572 B Ruger Street
San Francisco, CA 94129-0920

T 415.561.6276
F 415.561.6120

www.isma.tv

Internet Streaming Media Alliance Hyperlinked Video Specification

Version 1.0

September 2006

ISMA SPECIFICATION LIMITATIONS AND CONDITIONS OF USE

LEGAL LIMITATIONS AND CONDITIONS OF USE

USERS OF THE ISMA SPECIFICATION ARE NOT PERMITTED OR AUTHORIZED TO STATE OR CLAIM THAT THEIR PRODUCTDS OR APPLICATIONS COMPLY WITH THE SPECIFICATION, PENDING ISMA'S DEVELOPMENT AND IMPLEMENTATION OF A COMPLIANCE OR CERTIFICATION PROGRAM AND USER'S EXPRESS AGREEMENT WITH THE TERMS AND CONDITIONS THEREOF. BY REQUESTING OR USING THE SPECIFICATION, USER AGREES TO THIS LIMITATION AND CONDITION.

Table Of Contents

Document Status

This version of the document is version 1.0 of the ISMA URL-Streams Specification as proposed for *Public Review*. It is approved by the Technical Committee and by the ISMA Board of Directors.

1 Acronyms & Terms

AU	Access Unit – smallest media sample with timing information
AV	Audio-Visual
HTML	Hyper Text Markup Language [15]
HTTP	Hyper Text Transport Protocol [16]
IETF	Internet Engineering Task Force
ISMA	Internet Streaming Media Alliance
ISO	International Standards Organization
MP4	MPEG-4 File Format [10]
RFC	Request For Comments
RTP	Real time Transport Protocol [11]
RTSP	Real time Streaming Protocol [13]
SDP	Session Description Protocol [14]
URL	Uniform Resource Locator [6]
URI	Uniform Resource Identifier [6]

2 Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].

3 Scope

This technical specification describes a simple and light weight approach to augment audiovisual streaming and file playback with synchronized graphics and data. The data format and rendering process for the graphical data is not covered by this specification. However, HTML documents [15] which are accessed over HTTP [16] and rendered in a browser application are considered as an example of particular importance.

The main function required by a media client conforming to this specification is to *dispatch* Uniform Resource Locators (URLs) to an associated application (e.g. HTML browser) and thereby trigger an appropriate action (e.g. load and render an HTML document). URLs are dispatched at a specific point in time and may either be dispatched automatically or by user interaction. An example for the first case is of a lecture, where the dispatched URL causes a separate browser-frame, or window, to load slides as a talk proceeds. An example of the second case is an instructional video or advertisement where some frames say "for more information on this, click now, and you will be taken to a suitable page".

Besides the expected behavior when dispatching URLs, this specification describes the Access Unit (AU) format for URLs and how these are stored in MP4 files and streamed over RTP. In the latter case also the signaling in SDP is covered.

This specification is an *ISMA Technical Specification*, meaning that it provides a building block for higher level System Specifications. Such system specifications may also define other data formats to be referenced by URLs or define the exact HTML version support and profile (CSS, JavaScript, etc.). Considering audiovisual media, this specification is particularly suited to be combined with Version 1.0 and 2.0 of the ISMA specification [2,3] but can also be used with other specifications or stand-alone.

4 Architecture and Scenarios

In the following we assume that the component responsible for rendering graphical data is an HTML browser. Consequently, URLs will point to HTML documents which are available from an HTTP server. Other data formats and external applications can be supported as well. However, for the given example the basic architecture components include

- the AV Server,
- the IP Network,
- the AV Client,
- the HTTP Server,
- the HTML Browser,

as illustrated in Fig. 1. Note that a number of intermediate systems may be included in the transmission chain (storage, transcoders, caches/proxies) and that some components are only needed when content is accessed remotely. The AV Client receives or reads (1a/b) the audiovisual (AV) content, decodes, and renders it. In addition, it dispatches the received (or read) URLs to the HTML Browser (2) which acts as the rendering component. The browser loads the requested HTML document and renders it (3,4). The AV Client and the HTML Browser act in concert to provide the overall functionality but may be implemented in several ways (e.g. player with external browser application, or browser with embedded player plugin, or stand-alone rich media player). This specification only covers the expected behavior of the AV Client, and in particular when and how URLs are dispatched.

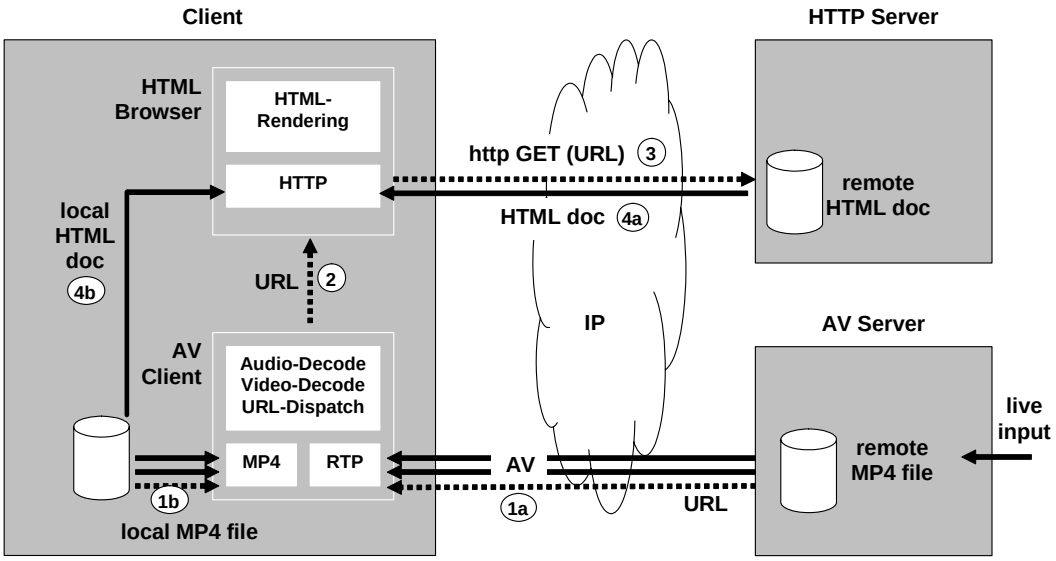


Figure 1: URL-Streams Architecture Overview

The AV Client may obtain the AV content and URLs through RTP streaming (1a) or from a local MP4 file (1b). In case of RTP streaming, the AV Server may encode the content in real-time or read it from file. Furthermore, also the HTML content may be stored on a remote HTTP server (4a) or locally (4b). These options can be combined in many useful playback scenarios, most of which are supported by this specification (one exception being that e.g. a URL streamed via RTP cannot reference local HTML content).

5 General

URLs are considered a separate media similar to audio and video. Hence, a separate track is used to store the URLs in an MP4 file and a separate RTP-stream is used to stream the URLs over an IP network. Section 5.1 describes the Access Unit (AU) format for URLs. I.e., in the same way as e.g. an audio codec defines the format of a compressed audio frame, this section describes how a URL is encoded and what syntax and semantic is used. The same AU format is used for storage (Section 6) as well as for streaming (Section 7).

Timing: As other media, URL-streams have a temporal aspect. The timestamp of an AU is the time that the URL is being dispatched or is becoming active. When streamed, this is the RTP timestamp. When stored in file, the timestamp of an AU is obtained by adding the duration of all preceding AUs (as given in the time-to-sample table 'stts').

Auto-Dispatch vs. Click-Through URLs: A URL may either be dispatched automatically at the specified media time (*auto-dispatch URL*) or it may be dispatched depending on some user interaction, typically a mouse click on the video region (*click-through URL*). In this case, the timestamp and duration of the AU specifies the time window when the URL is *active*. Hence, if the user interacts during this time window, the click-through URL is dispatched. Outside the active time window, or without any user interaction, the URL is not dispatched at all. The provision of a 'place to click' when the content-stream is audio-only, or otherwise has no visual representation, is terminal dependent.

Targets: A URL may specify a target, similar to the target attribute in an HTML hyperlink (<a>). In this case, the URL shall be dispatched to the specified target. One common example for targets are frames within a browser, where different URLs are dispatched to different frames (e.g. one frame for the slides of a talk and another frame for the biography of the speaker).

Rendering: The URLs within the AUs are not required to be user-visible. They perform, or enable the user to perform, a control operation. User software may display the URLs or not, as it chooses. There is no normative location or form of the rendering, if any. Just as browsers display tool-tip hints etc. that show when something may happen if a click occurs, user agents are encouraged to show the user that there is the possibility of an active 'click' when a click-through URL is active, and likewise may show the available or dispatched URLs in some system-specific way.

Conformance: A conformant implementation is one that correctly dispatches the specified URLs into the specified target. The content of the URL is not constrained by this specification (except for basic syntax rules, see Section 5.1.1) and there is no conformance statement with respect to the protocol used (i.e. the scheme before the first colon in an absolute URL, thus "http://") or the content-type returned. It is the author's responsibility to deliver content into an environment where both the target and the URL can be interpreted.

5.1 Access Unit Format

Each AU in the media data consists of a text string. The size of the AU is the number of bytes in the string. There is no need for null termination of the text string. Authors should limit the string in each AU to no more than 1408 bytes, for maximum terminal interoperability.

The characters in the string must be within the US-ASCII character set and are encoded in US-ASCII [4]. The string is formed as a series of elements, consisting of single uppercase letters identifying the type of the element, followed by the body of the element enclosed in angle-brackets thus: "<>". The high-level syntax based on ABNF [5] is given by

```
url-au = url-elmt [target-elmt] [location-elmt]

url-elmt      = ["A"] "<" [url] ">"
target-elmt   = "T" "<" target ">"
location-elmt = "M<>"
```

which is described further below (note that the above ABNF does not fully define the syntax and is therefore not normative). Examples for typical URL AUs include:

```
A<>
A<http://www.server.com/dir1/doc2.html>
<http://www.server.com/dir1/doc2.html>T<_blank>
<http://www.server.com/dir1/doc2.html>T<_top>M<>
<doc1.html>T<frame1>
<rtsp://www.av.com/cnn.sdp>T<_stream>
```

The semantics are described in more detail in Section 5.1.1 – 5.1.3. In order to assure backwards compatibility with future versions of this specification, AV Clients shall skip unknown elements in the AU gracefully. For example

`<doc1.html>T<frame1>` and

`<doc1.html>`**`E<some-extension>`**`T<frame1>`

shall be treated identical by an AV Client complying to Version 1.0 of this specification.

5.1.1 The URL Element

The URL element is always the first element of an AU and is mandatory. If the upper-case letter A precedes the body, the URL is automatically dispatched at the specified time. If the string starts with the body (i.e. the character "<"), it is made available as a click-through URL, i.e. it is only dispatched after user interaction.

The body of the URL element includes the URL. It shall comply to RFC 3986 [6], which builds URLs from the US-ASCII character set. It is recommended to avoid characters outside the US-ASCII character set for resources (e.g. http://www.server.com/müller.html). However, if required, such a resource can be encoded into a valid URL by using percent-encoding of an extended ASCII character set (e.g. http://www.server.com/m%FCller.html assuming ISO Latin-9 [7]) or other character encoding schemes (e.g. http://www.server.com/m%C3%BCller.html assuming UTF-8 [8]). This specification does not define a specific character encoding scheme. It is the responsibility of the content author to assure that the used encoding scheme in the URL allows retrieving a resource from the specified server.

The URL can be absolute or relative. Relative URLs may only be used if a Base URL is defined (see Section 5.2). Relative URLs are transformed to absolute URLs by merging the relative URL with the Base URL according to RFC 3986 [6].

Empty URL: The body of an URL element may also be empty (“<>”). This special URL may be used in two cases:

a) At the beginning of an URL track: It is possible that the beginning of a URL stream (i.e. the timestamp when the first URL shall become active) is not equal to the beginning of the presentation (timestamp = 0). This can be achieved by inserting an empty URL as the first AU in the URL track. URL-stream capable players encountering an empty URL must not open a browser window and must not change the content of an already open browser window.
b) After a click-through URL: In this case an empty URL disables the last click-through URL.

The content referenced by a URL shall be *displayed* until the next auto-dispatch URL for the same target (see Section 5.1.2) is dispatched or an active click-through URL for the same target is triggered by the user. A click-through URL shall be active until one of the following three events occurs: 1) an auto-dispatch URL for the same target is dispatched, 2) a new click-through URL is becoming active, or 3) an empty URL is dispatched. Note that there is only one active click-through URL at each point in time.

Informative Note: Empty URLs are not intended to blank or close frames or external browser windows. Depending on the environment, this can be achieved by e.g. sending the URL "<about:blank>" or "<javascript:window.close()>" respectively.

5.1.2 The Target Element

A target element may follow the URL element. If it exists, the body is preceded by an upper-case T. Permitted target names include:

a) the special target <_stream>, which identifies the calling media stream (i.e. the currently playing audio/video is replaced with the content at the given URL),

b) the special strings used for targets in HTML (_top, _self, _blank, _parent),

c) the names of frames within the same browser context.

Note that the target <_stream> imposes constraints on the type of resource that is referenced by the URL. For example, an AV Client which is currently playing an RTP stream may ignore the AU "<http://www.server.com/dir1/doc2.html>T<_stream>" because it cannot handle HTML content. Hence, when using <_stream>, the resource which is referenced by the URL is limited to types that can be handled by the AV Client.

5.1.3 The Click-Location Element

The click-location element, if present, is preceded by an upper-case character M and has an empty body, thus “M<>”. If it occurs, the AV Client appends two parameters named "xclick" and "yclick" together with their values to the query string so that the resulting URL is valid according to the URI syntax of the scheme. So for example, <http://my.com/map.cgi>M<> might be dispatched as http://my.com/map.cgi?xclick=117&yclick=54, whereas <http://my.com/map.cgi?uid=42>M<> might be dispatched as http://my.com/map.cgi?uid=42&xclick=117&yclick=54. The location is relative to (that is, within) the video content of the source material. The coordinate system is from left-to-right (x) and top-to-bottom (y), i.e. the upper left corner has coordinates (0,0).

5.2 Base URL

This specification allows the definition of a *Base URL* for an URL-stream. A Base URL will e.g. allow changing the web server address without editing the media data (MDAT) of an MP4 file, so that a presentation can be moved from one server to another with only little editing effort. If a Base URL is defined, then URL Elements may contain relative URLs. Relative URLs are merged with the Base URL as described in RFC 3986 [6]. The Base URL must be valid according to RFC 3986. If no Base URL is defined only absolute URLs must be used in the stream. During a session, only a single Base URL is defined.

In the case of local file playback or progressive download, the Base URL can be defined explicitly by including the `HrefBaseUrlBox` into the sample table box ('stbl') of the MP4 file (see Section 6). However, if no `HrefBaseUrlBox` is present, then the default Base URL is given by the location of the MP4 file (if the content is stored in several MP4 files which are linked through references, then the Base URL is the location of the file containing the sample table box that defines the URL-stream). Hence, there is always a defined Base URL in the case of local file playback or progressive download – either explicitly or by default.

In the case of streaming, the Base URL can be defined in the SDP using the `base_url` parameter on the fmtp line (see Section 7). If it is not present in the SDP then it is undefined and only absolute URLs must be used in the URL-stream. Worded differently, a Base URL must be present in the SDP if relative URLs are used in the URL-stream.

6 Storage

For strorage of URL AUs, this specification builds on the ISMA 2.0 [3] architecture, including the ISO base file format [9] and the MP4 file format [10].

6.1 Media Handler

A URL stream is in a new class of stream. For all files in the ISO file format family, including the MP4 file format, the handler-type within the ‘hdlr’ box shall be ‘cntl’ (control).

6.2 Media Handler Header

The URL track uses an empty null media header (‘nmhd’), called Mpeg4MediaHeaderBox in the MP4 specification, in common with other MPEG streams:

```
aligned(8) class  Mpeg4MediaHeaderBox
    extends FullBox(’nmhd’, version = 0, flags) {
}
```

6.3 Sample description format

The sample table box ('stbl') contains sample descriptions for the URL track, in the sample description box ('stsd') as named sample entries. Each entry is a sample entry box of type ‘href’. This name defines the format both of the sample description and the samples associated with that sample description. Terminals shall not attempt to decode or display sample descriptions with unrecognized names, nor the samples attached to those sample descriptions.

The format starts with the standard fields (the reserved bytes and the data reference index). The HrefDescription extends the regular sample entry by adding the HrefBaseUrlBox. The HrefBaseUrlBox contains just the base URL string:

```
class HrefBaseUrlBox extends Box(‘burl’) {
    string base_url;
}
```

```
class HrefSampleEntry() extends SampleEntry (‘href’) {
    Box[]     reserved;
     HrefBaseUrlBox ();    // optional;
}
```

6.4 Storage of Samples

A sample in a file is an access unit (AU) as defined above.

In a file, the sample size table provides the complete byte-count of each sample. The size is indicated in other ways in other environments (e.g. RTP, see below).

The timestamp of the sample is defined by the decoding time to sample box (‘stts’).

7 Streaming

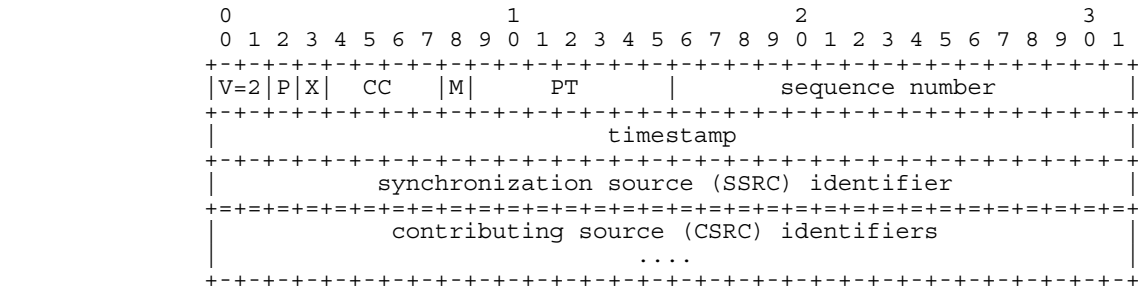
For streaming of URL AUs, this specification builds on the ISMA 2.0 [3] architecture, including RTP [11,12] for media transport, RTSP [13] for media control, and SDP [14] for media description. Since, URLs constitute a new media format, a specific payload format is defined in Section 7.1 for carriage over RTP. Signaling of URL streams based on SDP in covered in Section 7.2.

7.1 RTP Payload Format

A typical URL AU consists of roughly 100 bytes. This means that an RTP packet can comfortably carry several URL AUs. Hence, the defined payload format does allow carrying several AUs within one RTP packet, where each AU has its own associated timestamp (support for aggregation). However, AUs cannot be split over several RTP packets, i.e. only complete AUs can be carried (no support of fragmentation).

AUs may be repeated in the stream, for error resilience. Repeated AUs may be detected as they have the same timestamp. Two distinct (different) AUs must not have the same timestamp. Repetition is permitted here as a typical URL stream is of very low bandwidth (tens of bytes per second), yet the loss of an AU may lose important information.

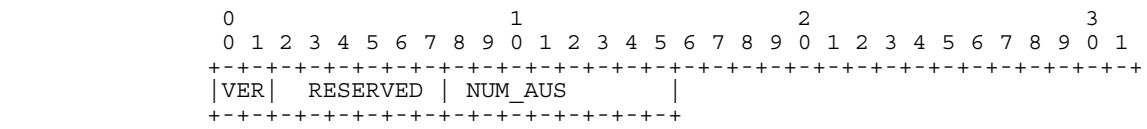
The RTP header has the following format:



Most of these parameters are defined in RFC 3550 [11]. We note that in our case:

```
P  = 0
X  = 0
M  = 1
PT = <dynamic payload type>
```
(as defined in SDP, see below)

The RTP header shall be followed by the following two bytes.



Here,

VER is the version number of this specification – 00 for now.

RESERVED is reserved for future use.

NUM_AUS holds the number of URL AUs in the payload.

The payload holds the URL AUs, sequentially, in the following format:

- 16 bits of data representing the length of the first AU in this payload.
- The first URL AU, represented as defined above.
- This is followed by NUM_AUS-1 repetitions of:
    - 16 bits of unsigned integer data representing the delta timestamp for the next AU. The effective timestamp of the AU is calculated by adding this field to the timestamp of the RTP packet. The delta timestamp is strictly positive; the AUs MUST be ordered in time order within the payload, and the delta must not be zero (i.e. repetition of the same AU within a packet is not permitted).
    - 16 bits of unsigned integer data representing the length of the next AU in this payload.
    - The subsequent URL AU, represented as defined above.

7.2 SDP Signaling

The mandatory SDP media section looks like:

```
m=control <port number>[/1] RTP/AVP <dynamic payload number>
a=rtpmap: <dynamic payload number> X-HREF/<clock rate>
```

In the above:

<port number> can be selected by the server.
<dynamic payload number> should be in the range 96..127.
<clock rate> indicates clock rate of the timestamps.

When a Base URL is used, the following additions are made:

```
a=fmtp: <dynamic payload type> base_url=<URL>
```

Additionally, for RTSP streams, a control URL should be supplied, as usual, for example

```
a=control:trackID=<id>
```

References

[1] IETF RFC 2119: “Key words for use in RFCs to Indicate Requirement Levels”, S. Bradner, March 1997.

[2] Internet Media Streaming Alliance, “Internet Streaming Media Alliance Implementation Specification, Version 1.0.1”, June 3, 2004.

[3] Internet Media Streaming Alliance, “Internet Streaming Media Alliance Implementation Specification, Version 2.0”, April, 2005.

[4] American National Standards Institute, "Coded Character Set -- 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986.

[5] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[6] IETF RFC 3986: “Uniform Resource Identifier (URU): Generic Syntax”, T. Berners-Lee et al., January 2005.

[7] ISO/IEC 8859-15, “Information technology – 8-bit single-byte coded graphic character sets -- Part 15: Latin alphabet No. 9”.

[8] IETF RFC 3629: “UTF-8, a transformation format of ISO 10646”, F. Yergeau, November 2003.

[9] ISO/IEC 14496-12:2003 | 15444-12:2003: “Information technology – Coding of audio-visual objects – Part 12: ISO base media file format” | “Information technology – JPEG 2000 image coding system – Part 12: ISO base media file format”.

[10] ISO/IEC 14496-14:2003: “Information technology – Coding of audio-visual objects – Part 14: MP4 file format”.

[11] IETF RFC 3550: “RTP: A Transport Protocol for Real-Time Applications”, Schulzrinne H. et al., July 2003.

[12] IETF RFC 3551: “RTP Profile for Audio and Video Conferences with Minimal Control”, Schulzrinne H. and Casner S., July 2003.

[13] IETF RFC 2326: “Real Time Streaming Protocol (RTSP)”, Schulzrinne H., Rao A. and Lanphier R., April 1998.

[14] IETF RFC 2327: “SDP: Session Description Protocol”, Handley M. and Jacobson V., April 1998.

[15] W3C Recommendation, "HTML 4.01 Specification", D. Raggett, 24 December 1999.

[16] IETF RFC 2616: “Hypertext Transfer Protocol – HTTP/1.1”, R. Fielding et al., June 1999.