SDP: Session Description Protocol

Status of this Memo

This document is a product of the Multiparty Multimedia Session Control
(MMUSIC) working group of the Internet Engineering Task Force. Comments
are solicited and should be addressed to the working group's mailing
list at confctrl@isi.edu and/or the authors.

                                Abstract

    This memo defines the Session Description Protocol (SDP). SDP
    is intended for describing multimedia sessions for the
    purposes of session announcement, session invitation, and
    other forms of multimedia session initiation.

1.   Introduction

On the Internet multicast backbone (Mbone), a session directory tool is
used to advertise multimedia conferences and communicate the conference
addresses and conference tool-specific information necessary for
participation. This document defines a session description protocol for
this purpose, and for general real-time multimedia session description
purposes. This draft does not describe multicast address allocation or
the distribution of SDP messages. These are described in accompanying
drafts. SDP is not intended for negotiation of media encodings.

2.   Background

The Mbone is the part of the internet that supports IP multicast, and
thus permits efficient many-to-many communication. It is used
extensively for multimedia conferencing. Such conferences usually have
the property that tight coordination of conference membership is not
necessary; to receive a conference, a user at an Mbone site only has to
know the conference's multicast group address and the UDP ports for the
conference data streams.

Session directories assist the advertisement of conference sessions and
communicate the relevant conference setup information to prospective
participants. SDP is designed to convey such information to recipients.
SDP is purely a format for session description - it does not incorporate
a transport protocol, and is intended to use different transport
protocols as appropriate including the Session Announcement Protocol
[4], Session Initiation Protocol [11], Real-Time Streaming Protocol
[12], electronic mail using the MIME extensions, and the Hypertext
Transport Protocol.

SDP is intended to be general purpose so that it can be used for a wider
range of network environments and applications than just multicast
session directories. However, it is not intended to support negotiation
of session content or media encodings - this is viewed as outside the
scope of session description.

3.   Glossary of Terms

The following terms are used in this document, and have specific meaning
within the context of this document.

Conference
     A multimedia conference is a set of two or more communicating users
     along with the software they are using to communicate.

Session
    A multimedia session is a set of multimedia senders and receivers
    and the data streams flowing from senders to receivers. A
    multimedia conference is an example of a multimedia session.

Session Advertisement
    See session announcement.

Session Announcement
    A session announcement is a mechanism by which a session description
    is conveyed to users in a pro-active fashion, i.e., the session
    description was not explicitly requested by the user.

Session Description
    A well defined format for conveying sufficient information to
    discover and participate in a multimedia session.

3.1. Terminology

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 [13].

4. Examples of SDP Usage

4.1. Session Initiation

The Session Initiation Protocol, SIP [11] is an application-layer
control protocol for creating, modifying and terminating sessions such
as Internet multimedia conferences, Internet telephone calls and
multimedia distribution. The SIP messages used to create sessions carry
session descriptions which allow participants to agree on a set of
compatible media types. These session descriptions are commonly
formatted using SDP.

4.2. Streaming media

The Real Time Streaming Protocol, RTSP [12], is an application-level
protocol for control over the delivery of data with real-time
properties. RTSP provides an extensible framework to enable controlled,
on-demand delivery of real-time data, such as audio and video. It is
necessary for RTSP to convey a description of the session to be
controlled: SDP is often used for this purpose.

4.3. Multicast Announcement

In order to assist the advertisement of multicast multimedia conferences
and other multicast sessions, and to communicate the relevant session
setup information to prospective participants, a distributed session
directory may be used. An instance of such a session directory
periodically sends packets containing a description of the session to a
well known multicast group. These advertisements are received by other
session directories such that potential remote participants can use the
session description to start the tools required to participate in the
session.

One protocol commonly used to implement such a distributed directory is
the Session Announcement Protocol, SAP [4]. SDP provides the recommended
session description format for such announcements.

4.4. Email and the World Wide Web

Alternative means of conveying session descriptions include electronic
mail and the World Wide Web. For both email and WWW distribution, the
use of the MIME content type ``application/sdp'' MUST be used. This
enables the automatic launching of applications for participation in the
session from the WWW client or mail reader in a standard manner.

Note that announcements of multicast sessions made only via email or the
World Wide Web (WWW) do not have the property that the receiver of a
session announcement can necessarily receive the session because the
multicast sessions may be restricted in scope, and access to the WWW
server or reception of email is possible outside this scope. SAP
announcements do not suffer from this mismatch.

5. Requirements and Recommendations

The purpose of SDP is to convey information about media streams in
multimedia sessions to allow the recipients of a session description to
participate in the session. SDP is primarily intended for use in an
internetwork, although it is sufficiently general that it can describe
conferences in other network environments.

A multimedia session, for these purposes, is defined as a set of media
streams that exist for some duration of time. Media streams can be
many-to-many. The times during which the session is active need not be
continuous.

Thus far, multicast based sessions on the Internet have differed from

many other forms of conferencing in that anyone receiving the traffic
can join the session (unless the session traffic is encrypted). In such
an environment, SDP serves two primary purposes. It is a means to
communicate the existence of a session, and is a means to convey
sufficient information to enable joining and participating in the
session. In a unicast environment, only the latter purpose is likely to
be relevant.

Thus SDP includes:

o   Session name and purpose

o   Time(s) the session is active

o   The media comprising the session

o   Information to receive those media (addresses, ports, formats and so
     on)

As resources necessary to participate in a session may be limited, some
additional information may also be desirable:

o   Information about the bandwidth to be used by the conference

o   Contact information for the person responsible for the session

In general, SDP must convey sufficient information to be able to join a
session (with the possible exception of encryption keys) and to announce
the resources to be used to non-participants that may need to know.

5.1. Media Information

SDP includes:

o   The type of media (video, audio, etc)

o   The transport protocol (RTP/UDP/IP, H.320, etc)

o   The format of the media (H.261 video, MPEG video, etc)

For an IP multicast session, the following are also conveyed:

o   Multicast address for media

o   Transport Port for media

This address and port are the destination address and destination port
of the multicast stream, whether being sent, received, or both.

For an IP unicast session, the following are conveyed:

o   Remote address for media

o   Transport port for contact address

The semantics of this address and port depend on the media and transport
protocol defined. By default, this is the remote address and remote
port to which data is sent, and the remote address and local port on
which to receive data. However, some media may define to use these to
establish a control channel for the actual media flow.

5.2. Timing Information

Sessions may either be bounded or unbounded in time. Whether or not
they are bounded, they may be only active at specific times.

SDP can convey:

o   An arbitrary list of start and stop times bounding the session

o   For each bound, repeat times such as "every Wednesday at 10am for
     one hour"

This timing information is globally consistent, irrespective of local
time zone or daylight saving time.

5.3. Private Sessions

It is possible to create both public sessions and private sessions. SDP
itself does not distinguish between these: private sessions are
typically conveyed by encrypting the session description during
distribution. The details of how encryption is performed are dependent
on the mechanism used to convey SDP – e.g. mechanisms are defined for
SDP transported using SAP [4] and SIP [11].

If a session announcement is private it is possible to use that private
announcement to convey encryption keys necessary to decode each of the
media in a conference, including enough information to know which
encryption scheme is used for each media.

5.4. Obtaining Further Information about a Session

A session description should convey enough information to decide whether
or not to participate in a session. SDP may include additional pointers
in the form of Universal Resources Identifiers (URIs) for more
information about the session.

5.5. Categorisation

When many session descriptions are being distributed by SAP, or any
other advertisement mechanism, it may be desirable to filter
announcements that are of interest from those that are not. SDP
supports a categorisation mechanism for sessions that is capable of
being automated.

5.6. Internationalization

The SDP specification recommends the use of the ISO 10646 character sets
in the UTF-8 encoding (RFC 2044) to allow many different languages to be
represented. However, to assist in compact representations, SDP also
allows other character sets such as ISO 8859-1 to be used when desired.
Internationalization only applies to free-text fields (session name and
background information), and not to SDP as a whole.

6. SDP Specification

SDP session descriptions are entirely textual using the ISO 10646
character set in UTF-8 encoding. SDP field names and attributes names
use only the US-ASCII subset of UTF-8, but textual fields and attribute
values may use the full ISO 10646 character set. The textual form, as
opposed to a binary encoding such as ASN/1 or XDR, was chosen to enhance
portability, to enable a variety of transports to be used (e.g, session
description in a MIME email message) and to allow flexible, text-based
toolkits (e.g., Tcl/Tk) to be used to generate and to process session
descriptions. However, since SDP may be used in environments where the
maximum permissable size of a session description is limited (e.g. SAP
announcements; SIP transported in UDP), the encoding is deliberately
compact. Also, since announcements may be transported via very
unreliable means (e.g., email) or damaged by an intermediate caching
server, the encoding was designed with strict order and formatting rules
so that most errors would result in malformed announcements which could
be detected easily and discarded. This also allows rapid discarding of
encrypted announcements for which a receiver does not have the correct
key.

An SDP session description consists of a number of lines of text of the
form
<type>=<value>
<type> is always exactly one character and is case-significant. <value>
is a structured text string whose format depends on <type>. It also
will be case-significant unless a specific field defines otherwise.
Whitespace MUST NOT be used either side of the '=' sign. In general
<value> is either a number of fields delimited by a single space
character or a free format string.

A session description consists of a session-level description (details
that apply to the whole session and all media streams) and optionally
several media-level descriptions (details that apply only to a single
media stream).

An announcement consists of a session-level section followed by zero or
more media-level sections. The session-level part starts with a 'v='
line and continues to the first media-level section. The media
description starts with an 'm=' line and continues to the next media
description or end of the whole session description. In general,
session-level values are the default for all media unless overridden by
an equivalent media-level value.

Some lines in each description are REQUIRED and some are OPTIONAL but
all MUST appear in exactly the order given here (the fixed order greatly
enhances error detection and allows for a simple parser). OPTIONAL
items are marked with a '*'.

Session description
     v= (protocol version)
     o= (owner/creator and session identifier).
     s= (session name)
     i=* (session information)
     u=* (URI of description)
     e=* (email address)
     p=* (phone number)
     c=* (connection information - not required if included in all media)
     b=* (bandwidth information)
        One or more time descriptions (see below)
     z=* (time zone adjustments)
     k=* (encryption key)
     a=* (zero or more session attribute lines)
        Zero or more media descriptions (see below)

Time description
     t= (time the session is active)
     r=* (zero or more repeat times)

Media description
     m= (media name and transport address)
     i=* (media title)
     c=* (connection information - optional if included at session-level)
     b=* (bandwidth information)
     k=* (encryption key)
     a=* (zero or more media attribute lines)

The set of 'type' letters is deliberately small and not intended to be
extensible -- an SDP parser MUST completely ignore any announcement that
contains a 'type' letter that it does not understand. The 'attribute'
mechanism ("a=" described below) is the primary means for extending SDP
and tailoring it to particular applications or media. Some attributes
(the ones listed in this document) have a defined meaning but others may
be added on an application-, media- or session-specific basis. An SDP
parser MUST ignore any attribute it doesn't understand.

The connection ('c=') and attribute ('a=') information in the session-
level section applies to all the media of that session unless overridden
by connection information or an attribute of the same name in the media
description. For instance, in the example below, each media behaves as
if it were given a 'recvonly' attribute.

An example SDP description is:

```
v=0
o=mhandley 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu (Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416 udp wb
a=orient:portrait
```

Text records such as the session name and information are bytes strings
which may contain any byte with the exceptions of 0x00 (Nul), 0x0a
(ASCII newline) and 0x0d (ASCII carriage return). The sequence CRLF
(0x0d0a) is used to end a record, although parsers should be tolerant
and also accept records terminated with a single newline character. By
default these byte strings contain ISO-10646 characters in UTF-8
encoding, but this default may be changed using the 'charset' attribute.

Protocol Version

v=0

The ''v='' field gives the version of the Session Description Protocol.
There is no minor version number.

Origin

o=<username> <session id> <version> <network type> <address type>
<address>

The ''o='' field gives the originator of the session (their username and
the address of the user's host) plus a session id and session version
number. <username> is the user's login on the originating host, or it
is ''-'' if the originating host does not support the concept of user
ids. <username> MUST NOT contain spaces. <session id> is a numeric
string such that the tuple of <username>, <session id>, <network type>,
<address type> and <address> form a globally unique identifier for the
session. The method of session id allocation is up to the creating
tool, but it has been suggested that a Network Time Protocol (NTP)
timestamp be used to ensure uniqueness [1]. <version> is a version
number for this announcement. It is needed for proxy announcements to
detect which of several announcements for the same session is the most

recent. Again its usage is up to the creating tool, so long as
<version> is increased when a modification is made to the session data.
Again, it is RECOMMENDED (but not mandatory) that an NTP timestamp is
used. <network type> is a text string giving the type of network.
Initially ``IN'' is defined to have the meaning ``Internet''. <address
type> is a text string giving the type of the address that follows.
Initially ``IP4'' and ``IP6'' are defined. <address> is the globally
unique address of the machine from which the session was created. For
an address type of IP4, this is either the fully-qualified domain name
of the machine, or the dotted-decimal representation of the IP version 4
address of the machine. For an address type of IP6, this is either the
fully-qualified domain name of the machine, or the compressed textual
representation of the IP version 6 address of the machine. For both IP4
and IP6, the fully-qualified domain name is the form that SHOULD be
given unless this is unavailable, in which case the globally unique
address may be substituted. A local IP address MUST NOT be used in any
context where the SDP description might leave the scope in which the
address is meaningful.

In general, the ``o='' field serves as a globally unique identifier for
this version of this session description, and the subfields excepting
the version taken together identify the session irrespective of any
modifications.

Session Name

s=<session name>

The ``s='' field is the session name. There MUST be one and only one
``s='' field per session description. The ``s='' field MUST NOT be empty
and SHOULD contain ISO 10646 characters (but see also the `charset'
attribute below). If a session has no meaningful name, the value ``s= ''
SHOULD be used (i.e. a single space as the session name).

Session and Media Information

i=<session description>

The ``i='' field is information about the session. There may be at most
one session-level ``i='' field per session description, and at most one
``i='' field per media. Although it may be omitted, this is NOT
RECOMMENDED for session announcements, and user interfaces for composing
sessions should require text to be entered. If it is present it must
contain ISO 10646 characters (but see also the `charset' attribute
below).

A single ``i='' field can also be used for each media definition. In
media definitions, ``i='' fields are primarily intended for labeling

media streams. As such, they are most likely to be useful when a single
session has more than one distinct media stream of the same media type.
An example would be two different whiteboards, one for slides and one
for feedback and questions.

URI

u=<URI>

o   A URI is a Universal Resource Identifier as used by WWW clients

o   The URI should be a pointer to additional information about the
     conference

o   This field is OPTIONAL, but if it is present it MUST be specified
     before the first media field

o   No more than one URI field is allowed per session description

Email Address and Phone Number

e=<email address>
p=<phone number>

o   These specify contact information for the person responsible for the
     conference. This is not necessarily the same person that created
     the conference announcement.

o   Inclusion of an email address or phone number is OPTIONAL. Note
     that the previous version of SDP specified that either an email
     field or a phone field MUST be specified, but this was widely
     ignored. The change brings the specification into line with common
     usage.

o   If these are present, they should be specified before the first
     media field.

o   More than one email or phone field can be given for a session
     description.

o   Phone numbers should be given in the conventional international
     format - preceded by a ``+'' and the international country code.
     There must be a space or a hyphen (``-'') between the country code
     and the rest of the phone number. Spaces and hyphens may be used to

split up a phone field to aid readability if desired. For example:

    p=+44-171-380-7777   or   p=+1 617 253 6011

o   Both email addresses and phone numbers can have an optional free
    text string associated with them, normally giving the name of the
    person who may be contacted. This should be enclosed in parenthesis
    if it is present. For example:

        e=mjh@isi.edu (Mark Handley)

    The alternative RFC822 name quoting convention is also allowed for
    both email addresses and phone numbers. For example,

        e=Mark Handley <mjh@isi.edu>

    The free text string should be in the ISO-10646 character set with
    UTF-8 encoding, or alternatively in ISO-8859-1 or other encodings if
    the appropriate charset session-level attribute is set.

Connection Data

c=<network type> <address type> <connection address>

The ``c='' field contains connection data.

A session announcement MUST contain either one ``c='' field in each
media description (see below) or a ``c='' field at the session-level.
It MAY contain a session-level ``c='' field and one additional ``c=''
field per media description, in which case the per-media values override
the session-level settings for the relevant media.

The first sub-field is the network type, which is a text string giving
the type of network. Initially ``IN'' is defined to have the meaning
``Internet''.

The second sub-field is the address type. This allows SDP to be used
for sessions that are not IP based. Currently only IP4 and IP6 are
defined.

The third sub-field is the connection address. Optional extra sub-
fields may be added after the connection address depending on the value
of the <address type> field.

For IP4 and IP6 addresses, the connection address is defined as follows:

o   If the session is multicast, the connection address will be an IP
    multicast group address. If the conference is not multicast, then

the connection address contains the unicast IP address of the
expected data source or data relay or data sink as determined by
additional attribute fields. It is not expected that unicast
addresses will be given in a session description that is
communicated by a multicast announcement, though this is not
prohibited.

o   Conferences using an IPv4 multicast connection address MUST also
     have a time to live (TTL) value present in addition to the multicast
     address. The TTL and the address together define the scope with
     which multicast packets sent in this conference will be sent. TTL
     values MUST be in the range 0-255.

     The TTL for the session is appended to the address using a slash as
     a separator. An example is:

          c=IN IP4 224.2.36.42/127

     IPv6 multicast does not use TTL scoping, and hence the TTL value
     MUST NOT be present for IPv6 multicast. It is expected that IPv6
     scoped addresses will be used to limit the scope of conferences.

     Hierarchical or layered encoding schemes are data streams where the
     encoding from a single media source is split into a number of
     layers. The receiver can choose the desired quality (and hence
     bandwidth) by only subscribing to a subset of these layers. Such
     layered encodings are normally transmitted in multiple multicast
     groups to allow multicast pruning. This technique keeps unwanted
     traffic from sites only requiring certain levels of the hierarchy.
     For applications requiring multiple multicast groups, we allow the
     following notation to be used for the connection address:

          <base multicast address>[/<ttl>]/<number of addresses>

     If the number of addresses is not given it is assumed to be one.
     Multicast addresses so assigned are contiguously allocated above the
     base address, so that, for example:

          c=IN IP4 224.2.1.1/127/3

     would state that addresses 224.2.1.1, 224.2.1.2 and 224.2.1.3 are to
     be used at a ttl of 127. This is semantically identical to
     including multiple ``c='' lines in a media description:

```
     c=IN IP4 224.2.1.1/127
     c=IN IP4 224.2.1.2/127
     c=IN IP4 224.2.1.3/127
```

Similarly, an IPv6 example would be:

```
     c=IN IP6 FF15::101/3
```

which is semantically equivalent to:

```
     c=IN IP6 FF15::101
     c=IN IP6 FF15::102
     c=IN IP6 FF15::103
```

(remembering that the TTL field is not present in IPv6 multicast).

Multiple addresses or ``c='' lines can only be specified on a per-media basis, and not for a session-level ``c='' field.

The slash notation described above MUST NOT be used for IP unicast addresses.

Bandwidth

b=<modifier>:<bandwidth-value>

o   This specifies the proposed bandwidth to be used by the session or media, and is OPTIONAL.

o   <bandwidth-value> is in kilobits per second by default. Modifiers may specify that alternative units are to be used (the modifiers defined in this memo use the default units).

o   <modifier> is a single alphanumeric word giving the meaning of the bandwidth figure.

o   Two modifiers are initially defined:

CT   Conference Total: If the bandwidth of a session or media in a session is different from the bandwidth implicit from the scope, a `b=CT:...' line should be supplied for the session giving the proposed upper limit to the bandwidth used. The primary purpose of this is to give an approximate idea as to whether two or more sessions can co-exist simultaneously.

AS   Application-Specific Maximum: The bandwidth is interpreted to be application-specific, i.e., will be the application's concept of

     maximum bandwidth. Normally this will coincide with what is set
     on the application's ''maximum bandwidth'' control if applicable.
     For RTP based applications, AS gives the RTP ''session bandwidth''
     as defined in section 6.2 of [2].

   Note that CT gives a total bandwidth figure for all the media at all
   sites. AS gives a bandwidth figure for a single media at a single
   site, although there may be many sites sending simultaneously.

o   Extension Mechanism: Tool writers can define experimental bandwidth
     modifiers by prefixing their modifier with ''X-''. For example:

        b=X-YZ:128

   SDP parsers MUST ignore bandwidth fields with unknown modifiers.
   Modifiers MUST be alpha-numeric and, although no length limit is
   given, they are recommended to be short.

Times, Repeat Times and Time Zones

t=<start time> <stop time>

o   ''t='' fields specify the start and stop times for a session.
     Multiple ''t='' fields MAY be used if a session is active at
     multiple irregularly spaced times; each additional ''t='' field
     specifies an additional period of time for which the session will be
     active. If the session is active at regular times, an ''r='' field
     (see below) should be used in addition to and following a ''t=''
     field - in which case the ''t='' field specifies the start and stop
     times of the repeat sequence.

o   The first and second sub-fields give the start and stop times for
     the session respectively. These values are the decimal
     representation of Network Time Protocol (NTP) time values in seconds
     [1]. To convert these values to UNIX time, subtract decimal
     2208988800.

     NTP timestamps are 64 bit values which wrap sometime in the year
     2036. Since SDP uses an arbitrary length decimal representation,
     this should not cause an issue (SDP timestamps will continue
     counting seconds since 1900, NTP will use the value modulo the 64
     bit limit).

o   If the stop-time is set to zero, then the session is not bounded,
     though it will not become active until after the start-time. If the
     start-time is also zero, the session is regarded as permanent.

     User interfaces SHOULD strongly discourage the creation of unbounded

and permanent sessions as they give no information about when the
session is actually going to terminate, and so make scheduling
difficult.

The general assumption may be made, when displaying unbounded
sessions that have not timed out to the user, that an unbounded
session will only be active until half an hour from the current time
or the session start time, whichever is the later. If behaviour
other than this is required, an end-time should be given and
modified as appropriate when new information becomes available about
when the session should really end.

Permanent sessions may be shown to the user as never being active
unless there are associated repeat times which state precisely when
the session will be active. In general, permanent sessions SHOULD
NOT be created for any session expected to have a duration of less
than 2 months, and should be discouraged for sessions expected to
have a duration of less than 6 months.

r=<repeat interval> <active duration> <list of offsets from start-time>

o   ``r='' fields specify repeat times for a session. For example, if
      a session is active at 10am on Monday and 11am on Tuesday for one
      hour each week for three months, then the <start time> in the
      corresponding ``t='' field would be the NTP representation of 10am
      on the first Monday, the <repeat interval> would be 1 week, the
      <active duration> would be 1 hour, and the offsets would be zero and
      25 hours. The corresponding ``t='' field stop time would be the NTP
      representation of the end of the last session three months later. By
      default all fields are in seconds, so the ``r='' and ``t='' fields
      might be:

          t=3034423619 3042462419
          r=604800 3600 0 90000

      To make description more compact, times may also be given in units
      of days, hours or minutes. The syntax for these is a number
      immediately followed by a single case-sensitive character.
      Fractional units are not allowed - a smaller unit should be used
      instead. The following unit specification characters are allowed:

          d - days (86400 seconds)
          h - hours (3600 seconds)
          m - minutes (60 seconds)
          s - seconds (allowed for completeness but not recommended)

Thus, the above announcement could also have been written:

     r=7d 1h 0 25h

Monthly and yearly repeats cannot currently be directly specified
with a single SDP repeat time - instead separate "t" fields should
be used to explicitly list the session times.

z=<adjustment time> <offset> <adjustment time> <offset> ....

o   To schedule a repeated session which spans a change from daylight-
     saving time to standard time or vice-versa, it is necessary to
     specify offsets from the base repeat times. This is required because
     different time zones change time at different times of day,
     different countries change to or from daylight time on different
     dates, and some countries do not have daylight saving time at all.

     Thus in order to schedule a session that is at the same time winter
     and summer, it must be possible to specify unambiguously by whose
     time zone a session is scheduled. To simplify this task for
     receivers, we allow the sender to specify the NTP time that a time
     zone adjustment happens and the offset from the time when the
     session was first scheduled. The ''z'' field allows the sender to
     specify a list of these adjustment times and offsets from the base
     time.

     An example might be:

     z=2882844526 -1h 2898848070 0

     This specifies that at time 2882844526 the time base by which the
     session's repeat times are calculated is shifted back by 1 hour, and
     that at time 2898848070 the session's original time base is
     restored. Adjustments are always relative to the specified start
     time - they are not cumulative.

o   If a session is likely to last several years, it is expected that
     the session announcement will be modified periodically rather than
     transmit several years worth of adjustments in one announcement.

Encryption Keys

k=<method>
k=<method>:<encryption key>

o   The session description protocol MAY be used to convey encryption
    keys. A key field is permitted before the first media entry (in
    which case it applies to all media in the session), or for each
    media entry as required.

o   The format of keys and their usage is outside the scope of this
    document, but see [3].

o   The method indicates the mechanism to be used to obtain a usable key
    by external means, or from the encoded encryption key given. The
    following methods are defined:

    k=clear:<encryption key>
        The encryption key (as described in [3] for RTP media streams
        under the AV profile) is included untransformed in this key
        field.

    k=base64:<encoded encryption key>
        The encryption key (as described in [3] for RTP media streams
        under the AV profile) is included in this key field but has been
        base64 encoded because it includes characters that are
        prohibited in SDP.

    k=uri:<URI to obtain key>
        A Universal Resource Identifier as used by WWW clients is
        included in this key field. The URI refers to the data
        containing the key, and may require additional authentication
        before the key can be returned. When a request is made to the
        given URI, the MIME content-type of the reply specifies the
        encoding for the key in the reply. The key should not be
        obtained until the user wishes to join the session to reduce
        synchronisation of requests to the WWW server(s).

    k=prompt
        No key is included in this SDP description, but the session or
        media stream referred to by this key field is encrypted. The
        user should be prompted for the key when attempting to join the
        session, and this user-supplied key should then be used to
        decrypt the media streams.

Attributes

a=<attribute>
a=<attribute>:<value>

Attributes are the primary means for extending SDP. Attributes may be
defined to be used as "session-level" attributes, "media-level"

attributes, or both.

A media description may have any number of attributes (``a='' fields)
which are media specific. These are referred to as "media-level"
attributes and add information about the media stream. Attribute fields
can also be added before the first media field; these "session-level"
attributes convey additional information that applies to the conference
as a whole rather than to individual media; an example might be the
conference's floor control policy.

Attribute fields may be of two forms:

o   property attributes. A property attribute is simply of the form
     ``a=<flag>''. These are binary attributes, and the presence of the
     attribute conveys that the attribute is a property of the session.
     An example might be ``a=recvonly''.

o   value attributes. A value attribute is of the form
     ``a=<attribute>:<value>''. An example might be that a whiteboard
     could have the value attribute ``a=orient:landscape''

Attribute interpretation depends on the media tool being invoked. Thus
receivers of session descriptions should be configurable in their
interpretation of announcements in general and of attributes in
particular.

Attribute names MUST be in the US-ASCII subset of ISO-10646/UTF-8.

Attribute values are byte strings, and MAY use any byte value except
0x00 (Nul), 0x0A (LF), and 0x0D (CR). By default, attribute values are
to be interpreted as in ISO-10646 character set with UTF-8 encoding.
Unlike other text fields, attribute values are NOT normally affected by
the 'charset' attribute as this would make comparisons against known
values problematic. However, when an attribute is defined, it can be
defined to be charset-dependent, in which case it's value should be
interpreted in the session charset rather than in ISO-10646.

Attributes that will be commonly used can be registered with IANA (see
Appendix B). Unregistered attributes should begin with "X-" to prevent
inadvertent collision with registered attributes. In either case, if an
attribute is received that is not understood, it should simply be
ignored by the receiver.

Media Announcements

m=<media> <port> <transport> <fmt list>

A session description may contain a number of media descriptions. Each
media description starts with an ''m='' field, and is terminated by
either the next ''m='' field or by the end of the session description.
A media field also has several sub-fields:

o   The first sub-field is the media type. Currently defined media are
     ''audio'', ''video'', ''application'', ''data'' and ''control'',
     though this list may be extended as new communication modalities
     emerge (e.g., telepresense). The difference between ''application''
     and ''data'' is that the former is a media flow such as whiteboard
     information, and the latter is bulk-data transfer such as
     multicasting of program executables which will not typically be
     displayed to the user. ''control'' is used to specify an additional
     conference control channel for the session.

o   The second sub-field is the transport port to which the media stream
     will be sent. The meaning of the transport port depends on the
     network being used as specified in the relevant ''c'' field and on
     the transport protocol defined in the third sub-field. Other ports
     used by the media application (such as the RTCP port, see [2])
     should be derived algorithmically from the base media port.

     Note: For transports based on UDP, the value should be in the range
     1024 to 65535 inclusive. For RTP compliance it SHOULD be an even
     number.

     For applications where hierarchically encoded streams are being sent
     to a unicast address, it may be necessary to specify multiple
     transport ports. This is done using a similar notation to that used
     for IP multicast addresses in the ''c='' field:

          m=<media> <port>/<number of ports> <transport> <fmt list>

     In such a case, the ports used depend on the transport protocol.
     For RTP, only the even ports are used for data and the corresponding
     one-higher odd port is used for RTCP. For example:

          m=video 49170/2 RTP/AVP 31

     would specify that ports 49170 and 49171 form one RTP/RTCP pair and
     49172 and 49173 form the second RTP/RTCP pair. RTP/AVP is the
     transport protocol and 31 is the format (see below).

     If multiple addresses are specified in the ''c='' field and multiple
     ports are specified in the ''m='' field, a one-to-one mapping from
     port to the corresponding address is implied. For example:

        c=IN IP4 224.2.1.1/127/2
        m=video 49170/2 RTP/AVP 31

   would imply that address 224.2.1.1 is used with ports 49170 and
   49171, and address 224.2.1.2 is used with ports 49172 and 49173.

o   The third sub-field is the transport protocol. The transport
     protocol values are dependent on the address-type field in the
     ``c='' fields.  Thus a ``c='' field of IP4 defines that the
     transport protocol runs over IP4. For IP4, it is normally expected
     that most media traffic will be carried as RTP over UDP. The
     following transport protocols are preliminarily defined, but may be
     extended through registration of new protocols with IANA:

     - RTP/AVP - the IETF's Realtime Transport Protocol using the
       Audio/Video profile carried over UDP.

     - udp - User Datagram Protocol

     If an application uses a single combined proprietary media format
     and transport protocol over UDP, then simply specifying the
     transport protocol as udp and using the format field to distinguish
     the combined protocol is recommended. If a transport protocol is
     used over UDP to carry several distinct media types that need to be
     distinguished by a session directory, then specifying the transport
     protocol and media format separately is necessary. RTP is an
     example of a transport-protocol that carries multiple payload
     formats that must be distinguished by the session directory for it
     to know how to start appropriate tools, relays, mixers or recorders.

     The main reason to specify the transport-protocol in addition to the
     media format is that the same standard media formats may be carried
     over different transport protocols even when the network protocol is
     the same - a historical example is vat PCM audio and RTP PCM audio.
     In addition, relays and monitoring tools that are transport-
     protocol-specific but format-independent are possible.

     For RTP media streams operating under the RTP Audio/Video Profile
     [3], the protocol field is ``RTP/AVP''. Should other RTP profiles
     be defined in the future, their profiles will be specified in the
     same way. For example, the protocol field ``RTP/XYZ'' would specify
     RTP operating under a profile whose short name is ``XYZ''.

o   The fourth and subsequent sub-fields are media formats. For audio
     and video, these SHOULD reference a MIME sub-type describing the
     format under the 'audio' and 'video' top-level MIME types.

When a list of payload formats is given, this implies that all of
these formats may be used in the session, but the first of these
formats is the default format for the session.

For media whose transport protocol is not RTP or UDP the format
field is protocol specific. Such formats should be defined in an
additional specification document.

For media whose transport protocol is RTP, SDP can be used to
provide a dynamic binding of media encoding to RTP payload type.
The encoding names in the RTP AV Profile do not specify unique audio
encodings (in terms of clock rate and number of audio channels), and
so they are not used directly in SDP format fields. Instead, the
payload type number should be used to specify the format for static
payload types and the payload type number along with additional
encoding information should be used for dynamically allocated
payload types.

An example of a static payload type is u-law PCM coded single
channel audio sampled at 8kHz. This is completely defined in the
RTP Audio/Video profile as payload type 0, so the media field for
such a stream sent to UDP port 49232 is:

    m=audio 49232 RTP/AVP 0

An example of a dynamic payload type is 16 bit linear encoded stereo
audio sampled at 16KHz. If we wish to use dynamic RTP/AVP payload
type 98 for such a stream, additional information is required to
decode it:

    m=video 49232 RTP/AVP 98
    a=rtpmap:98 L16/16000/2

The general form of an rtpmap attribute is:

    a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding paramete

For audio streams, <encoding parameters> may specify the number of
audio channels. This parameter may be omitted if the number of
channels is one provided no additional parameters are needed.
For video streams, no encoding parameters are currently specified.

Additional parameters may be defined in the future, but codec-
specific parameters should not be added. Parameters added to an
rtpmap attribute should only be those required for a session
directory to make the choice of appropriate media too to participate
in a session. Codec-specific parameters should be added in other
attributes.

Up to one rtpmap attribute can be defined for each media format
specified. Thus we might have:

```
m=audio 49230 RTP/AVP 96 97 98
a=rtpmap:96 L8/8000
a=rtpmap:97 L16/8000
a=rtpmap:98 L16/11025/2
```

RTP profiles that specify the use of dynamic payload types must
define the set of valid encoding names and/or a means to register
encoding names if that profile is to be used with SDP.

Experimental encoding formats can also be specified using rtpmap.
RTP formats that are not registered as standard format names must be
preceded by ``X-''. Thus a new experimental redundant audio stream
called GSMLPC using dynamic payload type 99 could be specified as:

```
m=audio 49232 RTP/AVP 99
a=rtpmap:99 X-GSMLPC/8000
```

Such an experimental encoding requires that any site wishing to
receive the media stream has relevant configured state in its
session directory to know which tools are appropriate.

Note that RTP audio formats typically do not include information
about the number of samples per packet. If a non-default (as
defined in the RTP Audio/Video Profile) packetisation is required,
the``ptime'' attribute is used as given below.

For more details on RTP audio and video formats, see [3].

o   Predefined formats for UDP protocol non-RTP media are as below.

Application Formats:

wb: LBL Whiteboard (transport: udp)

nt: UCL Network Text Editor (transport: udp)

Suggested Attributes

The following attributes are suggested. Since application writers may
add new attributes as they are required, this list is not exhaustive.

a=cat:<category>
    This attribute gives the dot-separated hierarchical category of the

     session. This is to enable a receiver to filter unwanted sessions
     by category. It would probably have been a compulsory separate
     field, except for its experimental nature at this time. It is a
     session-level attribute, and is not dependent on charset.

a=keywds:<keywords>
     Like the cat attribute, this is to assist identifying wanted
     sessions at the receiver. This allows a receiver to select
     interesting session based on keywords describing the purpose of the
     session. It is a session-level attribute. It is a charset dependent
     attribute, meaning that its value should be interpreted in the
     charset specified for the session description if one is specified,
     or by default in ISO 10646/UTF-8.

a=tool:<name and version of tool>
     This gives the name and version number of the tool used to create
     the session description. It is a session-level attribute, and is
     not dependent on charset.

a=ptime:<packet time>
     This gives the length of time in milliseconds represented by the
     media in a packet. This is probably only meaningful for audio data.
     It should not be necessary to know ptime to decode RTP or vat audio,
     and it is intended as a recommendation for the
     encoding/packetisation of audio. It is a media attribute, and is
     not dependent on charset.

a=maxptime:<maximum packet time>
     The maximum amount of media which can be encapsulated in each
     packet, expressed as time in milliseconds. The time shall be
     calculated as the sum of the time the media present in the packet
     represents. The time SHOULD be a multiple of the frame size. This is
     probably only meaningful for audio data. It is a media attribute,
     and is not dependent on charset. Note that this attribute was
     introduced after RFC 2327, and non updated implementations will
     ignore this attribute.

a=rtpmap:<payload type> <encoding name>/<clock rate>[/<encoding
     parameters>]
     See the section on Media Announcements (the ``m='' field). This may
     be either a session or media attribute.

a=recvonly
     This specifies that the tools should be started in receive-only mode
     where applicable. It can be either a session or media attribute, and
     is not dependent on charset. Note that recvonly applies to the media
     only, not to any associated control protocol (e.g. an RTP based
     system in recvonly mode SHOULD still send RTCP packets).

a=sendrecv
    This specifies that the tools should be started in send and receive
    mode. This is necessary for interactive conferences with tools such
    as wb which defaults to receive only mode. It can be either a
    session or media attribute, and is not dependent on charset.

a=sendonly
    This specifies that the tools should be started in send-only mode.
    An example may be where a different unicast address is to be used
    for a traffic destination than for a traffic source. In such a case,
    two media descriptions may be use, one sendonly and one recvonly. It
    can be either a session or media attribute, but would normally only
    be used as a media attribute, and is not dependent on charset. Note
    that sendonly applies only to the media, and any associated control
    protocol (e.g. RTCP) SHOULD still be received and processed as
    normal.

a=inactive
    This specifies that the tools should be started in inactive mode.
    This is necessary for interactive conferences where users can put
    other users on hold. No media is sent over an inactive media stream.
    Note that an RTP based system SHOULD still send RTCP, even if
    started inactive. It can be either a session or media attribute, and
    is not dependent on charset.

a=orient:<whiteboard orientation>
    Normally this is only used in a whiteboard media specification. It
    specifies the orientation of a the whiteboard on the screen. It is
    a media attribute. Permitted values are 'portrait', 'landscape' and
    'seascape' (upside down landscape). It is not dependent on charset

a=type:<conference type>
    This specifies the type of the conference. Suggested values are
    'broadcast', 'meeting', 'moderated', 'test' and 'H332'. 'recvonly'
    should be the default for 'type:broadcast' sessions, 'type:meeting'
    should imply 'sendrecv' and 'type:moderated' should indicate the use
    of a floor control tool and that the media tools are started so as
    to ''mute'' new sites joining the conference.

    Specifying the attribute type:H332 indicates that this loosely
    coupled session is part of a H.332 session as defined in the ITU
    H.332 specification [10]. Media tools should be started 'recvonly'.

    Specifying the attribute type:test is suggested as a hint that,
    unless explicitly requested otherwise, receivers can safely avoid
    displaying this session description to users.

    The type attribute is a session-level attribute, and is not

dependent on charset.

a=charset:<character set>
     This specifies the character set to be used to display the session
     name and information data. By default, the ISO-10646 character set
     in UTF-8 encoding is used. If a more compact representation is
     required, other character sets may be used such as ISO-8859-1 for
     Northern European languages. In particular, the ISO 8859-1 is
     specified with the following SDP attribute:

          a=charset:ISO-8859-1

     This is a session-level attribute; if this attribute is present, it
     must be before the first media field. The charset specified MUST be
     one of those registered with IANA, such as ISO-8859-1. The
     character set identifier is a US-ASCII string and MUST be compared
     against the IANA identifiers using a case-insensitive comparison.
     If the identifier is not recognised or not supported, all strings
     that are affected by it SHOULD be regarded as byte strings.

     Note that a character set specified MUST still prohibit the use of
     bytes 0x00 (Nul), 0x0A (LF) and 0x0d (CR). Character sets requiring
     the use of these characters MUST define a quoting mechanism that
     prevents these bytes appearing within text fields.

a=sdplang:<language tag>
     This can be a session level attribute or a media level attribute.
     As a session level attribute, it specifies the language for the
     session description. As a media level attribute, it specifies the
     language for any media-level SDP information field associated with
     that media. Multiple sdplang attributes can be provided either at
     session or media level if multiple languages in the session
     description or media use multiple languages, in which case the order
     of the attributes indicates the order of importance of the various
     languages in the session or media from most important to least
     important.

     In general, sending session descriptions consisting of multiple
     languages should be discouraged. Instead, multiple descriptions
     should be sent describing the session, one in each language.
     However this is not possible with all transport mechanisms, and so
     multiple sdplang attributes are allowed although not recommended.

     The sdplang attribute value must be a single RFC 1766 language tag
     in US-ASCII. It is not dependent on the charset attribute. An
     sdplang attribute SHOULD be specified when a session is of

     sufficient scope to cross geographic boundaries where the language
     of recipients cannot be assumed, or where the session is in a
     different language from the locally assumed norm.

a=lang:<language tag>
     This can be a session level attribute or a media level attribute.
     As a session level attribute, it specifies the default language for
     the session being described. As a media level attribute, it
     specifies the language for that media, overriding any session-level
     language specified. Multiple lang attributes can be provided either
     at session or media level if multiple languages if the session
     description or media use multiple languages, in which case the order
     of the attributes indicates the order of importance of the various
     languages in the session or media from most important to least
     important.

     The lang attribute value must be a single RFC 1766 language tag in
     US-ASCII. It is not dependent on the charset attribute. A lang
     attribute SHOULD be specified when a session is of sufficient scope
     to cross geographic boundaries where the language of recipients
     cannot be assumed, or where the session is in a different language
     from the locally assumed norm.

a=framerate:<frame rate>
     This gives the maximum video frame rate in frames/sec. It is
     intended as a recommendation for the encoding of video data.
     Decimal representations of fractional values using the notation
     "<integer>.<fraction>" are allowed. It is a media attribute, is
     only defined for video media, and is not dependent on charset.

a=quality:<quality>
     This gives a suggestion for the quality of the encoding as an
     integer value.

     The intention of the quality attribute for video is to specify a
     non-default trade-off between frame-rate and still-image quality.
     For video, the value in the range 0 to 10, with the following
     suggested meaning:

     10 - the best still-image quality the compression scheme can give.

     5   - the default behaviour given no quality suggestion.

     0   - the worst still-image quality the codec designer thinks is
           still usable.

It is a media attribute, and is not dependent on charset.

a=fmtp:<format> <format specific parameters>
    This attribute allows parameters that are specific to a particular
    format to be conveyed in a way that SDP doesn't have to understand
    them. The format must be one of the formats specified for the
    media. Format-specific parameters may be any set of parameters
    required to be conveyed by SDP and given unchanged to the media tool
    that will use this format.

    It is a media attribute, and is not dependent on charset.

6.1. Communicating Conference Control Policy

There is some debate over the way conference control policy should be
communicated. In general, the authors believe that an implicit
declarative style of specifying conference control is desirable where
possible.

A simple declarative style uses a single conference attribute field
before the first media field, possibly supplemented by properties such
as 'recvonly' for some of the media tools. This conference attribute
conveys the conference control policy. An example might be:

        a=type:moderated

In some cases, however, it is possible that this may be insufficient to
communicate the details of an unusual conference control policy. If
this is the case, then a conference attribute specifying external
control might be set, and then one or more ''media'' fields might be
used to specify the conference control tools and configuration data for
those tools. An example is an ITU H.332 session:

        ...
        c=IN IP4 224.5.6.7
        a=type:H332
        m=audio 49230 RTP/AVP 0
        m=video 49232 RTP/AVP 31
        m=application 12349 udp wb
        m=control 49234 H323 mc
        c=IN IP4 134.134.157.81

In this example, a general conference attribute (type:H332) is specified
stating that conference control will be provided by an external H.332
tool, and a contact addresses for the H.323 session multipoint
controller is given.

In this document, only the declarative style of conference control
declaration is specified. Other forms of conference control should
specify an appropriate type attribute, and should define the
implications this has for control media.

7. Security Considerations

SDP is a session description format that describes multimedia sessions.
A session description SHOULD NOT be trusted unless it has been obtained
by an authenticated transport protocol from a trusted source. Many
different transport protocols may be used to distribute session
description, and the nature of the authentication will differ from
transport to transport.

One transport that will frequently be used to distribute session
descriptions is the Session Announcement Protocol (SAP). SAP provides
both encryption and authentication mechanisms but due to the nature of
session announcements it is likely that there are many occasions where
the originator of a session announcement cannot be authenticated because
they are previously unknown to the receiver of the announcement and
because no common public key infrastructure is available.

On receiving a session description over an unauthenticated transport
mechanism or from an untrusted party, software parsing the session
should take a few precautions. Session description contain information
required to start software on the receivers system. Software that
parses a session description MUST not be able to start other software
except that which is specifically configured as appropriate software to
participate in multimedia sessions. It is normally considered
INAPPROPRIATE for software parsing a session description to start, on a
user's system, software that is appropriate to participate in multimedia
sessions, without the user first being informed that such software will
be started and giving their consent. Thus a session description
arriving by session announcement, email, sessioR multimedia,session page
SHOULD NOT deliver the user into an interactive
without the user being aware that this will happen. As it is not always
simple to tell whether a session is interactive or not, applications
that are unsure should assume sessions are interactive.

In this specification, there are no attributes which would allow the
recipient of a session description to be informed to start multimedia
tools in a mode where they default to transmitting. Under some
circumstances it might be appropriate to define such attributes. If
this is done an application parsing a session description containing
such attributes SHOULD either ignore them, or inform the user that
joining this session will result in the automatic transmission of
multimedia data. The default behaviour for an unknown attribute is to
ignore it.

Session descriptions may be parsed at intermediate systems such as
firewalls for the purposes of opening a hole in the firewall to allow
the participation in multimedia sessions. It is considered
INAPPROPRIATE for a firewall to open such holes for unicast data streams

unless the session description comes in a request from inside the
firewall. For multicast sessions, it is likely that local
administrators will apply their own policies, but the exclusive use of
"local" or "site-local" administrative scope within the firewall and the
refusal of the firewall to open a hole for such scopes will provide
separation of global multicast sessions from local ones.

Appendix A: SDP Grammar

This appendix provides an Augmented BNF grammar for SDP. ABNF is
defined in RFC 2234.

```
; SDP Syntax
announcement =          proto-version
                        origin-field
                        session-name-field
                        information-field
                        uri-field
                        email-fields
                        phone-fields
                        connection-field
                        bandwidth-fields
                        time-fields
                        key-field
                        attribute-fields
                        media-descriptions


proto-version =         "v=" 1*DIGIT CRLF
                        ;this memo describes version 0


origin-field =          "o=" username SP sess-id SP sess-version SP
                        nettype SP addrtype SP unicast-address CRLF


session-name-field =  "s=" text CRLF


information-field =   ["i=" text CRLF]


uri-field =           ["u=" uri CRLF]


email-fields =          *("e=" email-address CRLF)


phone-fields =          *("p=" phone-number CRLF)
```

```
connection-field =      ["c=" nettype SP addrtype SP
                        connection-address CRLF]
                        ;a connection field must be present
                        ;in every media description or at the
                        ;session-level


bandwidth-fields =      *("b=" bwtype ":" bandwidth CRLF)


time-fields =           1*( "t=" start-time SP stop-time
                        *(CRLF repeat-fields) CRLF)
                        [zone-adjustments CRLF]


repeat-fields =         "r=" repeat-interval SP typed-time
                        1*(SP typed-time)


zone-adjustments =      "z=" time SP ["-"] typed-time
                        *(SP time SP ["-"] typed-time)


key-field =             ["k=" key-type CRLF]


attribute-fields =      *("a=" attribute CRLF)


media-descriptions =    *( media-field
                        information-field
                        *connection-field
                        bandwidth-fields
                        key-field
                        attribute-fields )


media-field =           "m=" media SP port ["/" integer]
                        SP proto 1*(SP fmt) CRLF


; sub-rules of 'o='
username =              non-ws-string
                        ;pretty wide definition, but doesn't include space
```

```
sess-id =           1*DIGIT
                    ;should be unique for this originating username/host


sess-version =      1*DIGIT
                    ;0 is a new session


nettype =           token
                    ;typically "IN"


addrtype =          token
                    ;typically "IP4" or "IP6"


; sub-rules of 'u='
uri =               URI-reference; defined in RFC1630 and RFC2732


; sub-rules of 'e='
email-address =     email *SP "(" 1*email-safe ")" /
                    1*email-safe "<" email ">" /
                    email


email =             addr-spec ; defined in RFC2822
                              ; modified to remove CFWS


; sub-rules of 'p='
phone-number =      phone *SP "(" 1*email-safe ")" /
                    1*email-safe "<" phone ">" /
                    phone


phone =             "+" POS-DIGIT 1*(SP / "-" / DIGIT)
                    ;there must be a space or hyphen between the
                    ;international code and the rest of the number.

                    ; Should this use the tel: URL syntax?


; sub-rules of 'c='
connection-address = multicast-address / unicast-address
```

```
; sub-rules of 'b='
bwtype =            token


bandwidth =         1*DIGIT


; sub-rules of 't='
start-time =        time / "0"


stop-time =         time / "0"


time =              POS-DIGIT 9*DIGIT
                    ; 10-digit NTP time represents times between
                    ; 1931 and 5068 AD.  9* allows times after that
                    ; as well.


; sub-rules of 'r=' and 'z='
repeat-interval =   typed-time


typed-time =        POS-DIGIT *DIGIT [fixed-len-time-unit]


fixed-len-time-unit = "d" / "h" / "m" / "s"


; sub-rules of 'k='
key-type =          "prompt" /
                    "clear:" text /
                    "base64:" base64 /
                    "uri:" uri /
                    key-method [ ":" text ]


base64      =       *base64-unit [base64-pad]
base64-unit =       4base64-char
base64-pad  =       2base64-char "==" / 3base64-char "="
base64-char =       ALPHA / DIGIT / "+" / "/"


key-method =        token
```

```
; sub-rules of 'a='
attribute =          (att-field ":" att-value) / att-field


att-field =          token


att-value =          byte-string


; sub-rules of 'm='
media =              token
                     ;typically "audio", "video", "application"
                     ;or "data"


fmt =                token
                     ;typically an RTP payload type for audio
                     ;and video media


proto  =             token "/" token
                     / token
                     ;typically "RTP/AVP" or "udp" for IP4


port =               1*DIGIT
                     ;should in the range "1024" to "65535" inclusive
                     ;for UDP based media


; generic sub-rules: addressing
unicast-address =    IP4-address / IP6-address / FQDN / extension-addr


multicast-address =  IP4-multicast / IP6-multicast

IP4-multicast =      m1 3*( "." decimal-uchar )
                     "/" ttl [ "/" integer ]
                     ; IPv4 multicast addresses may be in the
                     ; range 224.0.0.0 to 239.255.255.255

m1 =                 ("22" ("4"/"5"/"6"/"7"/"8"/"9")) /
                     ("23" DIGIT ))

IP6-multicast =      hexpart [ "/" integer ]
                     ; IPv6 address starting with FF
```

```
ttl =                   (POS-DIGIT *2DIGIT) / "0"


FQDN =                  4*(alpha-numeric / "-" / ".")
                        ; fully qualified domain name as specified
                        ; in RFC1035


IP4-address =           b1 3*("." decimal-uchar) / "0.0.0.0"

b1 =                    decimal-uchar
                        ; less than "224"; not "0" or "127"

; The following is from RFC2373 Appendix B. It is a direct copy.
IP6-address =           hexpart [ ":" IP4-address ]

hexpart =               hexseq / hexseq "::" [ hexseq ] /
                        "::" [ hexseq ]

hexseq  =               hex4 *( ":" hex4)

hex4    =               1*4HEXDIG


; Generic for other address families
extension-addr =        non-ws-string


; generic sub-rules: datatypes
text =                  byte-string
                        ;default is to interpret this as IS0-10646 UTF8
                        ;ISO 8859-1 requires a "a=charset:ISO-8859-1"
                        ;session-level attribute to be used


byte-string =           1*(%x01-09/%x0B-0C/%x0E-FF)
                        ;any byte except NUL, CR or LF


non-ws-string =         1*(VCHAR/%x80-FF)
                        ;string of visible US-ASCII, or high-bit, characters
```

```
token-char =            %x21/%x23-27/%x2A-2B/%x2D-2E/%x30-39/%x41-5A/%x5E-7E
                        ; definition from RFC 2045 -
                        ; "any (US-ASCII) CHAR except SPACE, CTLs,
                        ; or tspecials".
                        ; the tspecials are ()<>@,;:


token =                 1*(token-char)


email-safe =            1*(%x01-09/%x0B-0C/%x0E-27/
                           %x2A-3B/%x3D/%x3E-FF)
                        ;any byte except NUL, CR, LF, or the quoting
                        ;characters ()<>


integer =               POS-DIGIT *DIGIT


; generic sub-rules: primitives
alpha-numeric =         ALPHA / DIGIT


POS-DIGIT =             %x31-39 ; 1 - 9


; external references:
; ALPHA, DIGIT, CRLF, SP, VCHAR: from RFC 2234
; URI-reference: from RFC1630 and RFC2732
; addr-spec: from RFC 2822
```

Appendix B: Guidelines for registering SDP names with IANA

There are seven field names that may be registered with IANA. Using the terminology in the SDP specification BNF, they are "media", "proto", "fmt", "att-field", "bwtype", "nettype" and "addrtype".

"media" (eg, audio, video, application, data).

> The set of media is intended to be small and not to be extended except under rare circumstances. The same rules should apply for media names as for top-level MIME content types, and where possible the same name should be registered for SDP as for MIME. For media other than existing MIME top-level content types, a standards-track RFC MUST be produced for a new top-level content type to be registered, and the registration MUST provide good justification why no existing media name is appropriate.

"proto"

> In general this should be an IETF standards-track transport protocol identifier such as RTP/AVP (rfc 1889 under the rfc 1890 profile).

> However, people will want to invent their own proprietary transport protocols. Some of these should be registered as a "fmt" using "udp" as the protocol and some of which probably can't be.

> Where the protocol and the application are intimately linked, such as with the LBL whiteboard wb which used a proprietary and special purpose protocol over UDP, the protocol name should be "udp" and the format name that should be registered is "wb". The rules for formats (see below) apply to such registrations.

> Where the proprietary transport protocol really carries many different data formats, it is possible to register a new protocol name with IANA. In such a case, an RFC MUST be produced describing the protocol and referenced in the registration. Such an RFC MAY be informational, although it is preferable if it is standards-track.

"fmt"

> The format namespace is dependent on the context of the "proto" field, so a format cannot be registered without specifying one or more transport protocols that it applies to.

> Formats cover all the possible encodings that might want to be transported in a multimedia session.

For RTP formats that have been assigned static payload types, the
payload type number is used. For RTP formats using a dynamic
payload type number, the dynamic payload type number is given as
the format and an additional "rtpmap" attribute specifies the
format and parameters.

For non-RTP formats, any unregistered format name may be
registered. If there is a suitable mapping from a MIME subtype to
the format, then the MIME subtype name should be registered. If
there is no suitable mapping from a MIME subtype, a new name should
be registered. In either case, unless there are strong reasons not
to do so, a standards-track RFC SHOULD be produced describing the
format and this RFC SHOULD be referenced in the registration.

"att-field" (Attribute names)

Attribute field names MAY be registered with IANA, although this is
not compulsory, and unknown attributes are simply ignored.

When an attribute is registered, it must be accompanied by a brief
specification stating the following:

o   contact name, email address and telephone number

o   attribute-name (as it will appear in SDP)

o   long-form attribute name in English

o   type of attribute (session level, media level, or both)

o   whether the attribute value is subject to the charset
       attribute.

o   a one paragraph explanation of the purpose of the attribute.

o   a specification of appropriate attribute values for this
       attribute.

IANA will not sanity check such attribute registrations except to
ensure that they do not clash with existing registrations.

Although the above is the minimum that IANA will accept, if the
attribute is expected to see widespread use and interoperability is
an issue, authors are encouraged to produce a standards-track RFC
that specifies the attribute more precisely.

Submitters of registrations should ensure that the specification is
in the spirit of SDP attributes, most notably that the attribute is

platform independent in the sense that it makes no implicit
assumptions about operating systems and does not name specific
pieces of software in a manner that might inhibit interoperability.

"bwtype" (bandwidth specifiers)

A proliferation of bandwidth specifiers is strongly discouraged.

New bandwidth specifiers may be registered with IANA. The
submission MUST reference a standards-track RFC specifying the
semantics of the bandwidth specifier precisely, and indicating when
it should be used, and why the existing registered bandwidth
specifiers do not suffice.

"nettype" (Network Type)

New network types may be registered with IANA if SDP needs to be
used in the context of non-internet environments. Whilst these are
not normally the preserve of IANA, there may be circumstances when
an Internet application needs to interoperate with a non-internet
application, such as when gatewaying an internet telephony call
into the PSTN. The number of network types should be small and
should be rarely extended. A new network type cannot be registered
without registering at least one address type to be used with that
network type. A new network type registration MUST reference an
RFC which gives details of the network type and address type and
specifies how and when they would be used. Such an RFC MAY be
Informational.

"addrtype" (Address Type)

New address types may be registered with IANA. An address type is
only meaningful in the context of a network type, and any
registration of an address type MUST specify a registered network
type, or be submitted along with a network type registration. A
new address type registration MUST reference an RFC giving details
of the syntax of the address type. Such an RFC MAY be
Informational. Address types are not expected to be registered
frequently.

Registration Procedure

To register a name the above guidelines should be followed regarding the
required level of documentation that is required. The registration
itself should be sent to IANA. Attribute registrations should include
the information given above. Other registrations should include the
following additional information:

o   contact name, email address and telephone number

o   name being registered (as it will appear in SDP)

o   long-form name in English

o   type of name ("media", "proto", "fmt", "bwtype", "nettype", or
        "addrtype")

o   a one paragraph explanation of the purpose of the registered name.

o   a reference to the specification (eg RFC number) of the registered
        name.

IANA may refer any registration to the IESG or to any appropriate IETF
working group for review, and may request revisions to be made before a
registration will be made.

Appendix C: Changes from RFC 2327

o   Clarify that a=recvonly does NOT mean that you don't send RTCP, and
       similarly for sendonly and inactive. These only effect the RTP
       stream.

o   Rewrite the ABNF syntax (thanks to Jonathan Lennox)

o   Add a=inactive attribute.

o   Add a=maxptime attribute.

o   RFC 2327 mandated that either e= or p= was required. Both are now
       optional, to reflect actual usage.

o   Removed references to "conference" from the description of the t=
       line, to make it less SAP oriented.

o   Note about wrap-around of NTP timestamps in t=

o   Update BNF to support IPv6.

o   References have been updated.

o   Section 3.1 was replaced with a reference to RFC 2119, and the memo
       has been updated to use the RFC 2119 terminology (MUST, SHOULD,
       etc).

o   Use of "application/sdp" as MIME a type for SDP files is now "MUST"
       rather than "SHOULD".

o   A number of sections have been updated to be less SAP specific, and
       to reference other current uses of SDP such as RTSP and SIP.

o   The section on concatenation of session descriptions (which was not
       allowed in SAP, but allowed in other cases) has been removed. It is
       assumed that transports of SDP specify will specify this.

o   The description of the c= line has been updated to reflect common
       usage of SDP, rather than Mbone conferencing with SAP.

o   The b= line no longer makes a normative reference to the Mbone FAQ
       for bandwidth limits at various TTLs. The AS modifier to b= is
       noted as being the RTP session bandwidth.

o   Define relation between the m= line and MIME types

o   Note use of s= in sessions with no meaningful name

o   Note that a=rtpmap is a media level attribute

Appendix D: Authors' Addresses

Mark Handley
AT&T Center for Internet Research at ICSI,
International Computer Science Institute,
1947 Center Street, Suite 600,
Berkeley, CA 94704, USA
Email: mjh@aciri.org

Van Jacobson
MS 46a-1121
Lawrence Berkeley Laboratory
Berkeley, CA 94720
United States
Email: van@ee.lbl.gov

Colin Perkins
USC Information Sciences Institute
3811 N. Fairfax Drive, Suite 200
Arlington, VA 22203
United States
Email: csp@isi.edu

References

[1] D. Mills, ``Network Time Protocol (version 3) specification and
    implementation", RFC 1305, March 1992.

[2] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, ``RTP: A
    Transport Protocol for Real-Time Applications'', RFC 1889, January
    1996.

[3] H. Schulzrinne, ``RTP Profile for Audio and Video Conferences with
    Minimal Control'', RFC 1890, January 1996.

[4] M. Handley, C. Perkins and E. Whelan, ``Session Announcement
    Protocol'', RFC 2974, October 2000.

[5] V. Jacobson and S. McCanne, ``vat - X11-based audio teleconferencing
    tool'' vat manual page, Lawrence Berkeley Laboratory, 1994.

[6] The Unicode Consortium, "The Unicode Standard -- Version 2.0",
    Addison-Wesley, 1996.

[7] ISO/IEC 10646-1:1993. International Standard -- Information
    technology -- Universal Multiple-Octet Coded Character Set (UCS)
    -- Part 1: Architecture and Basic Multilingual Plane. Five
    amendments and a technical corrigendum have been published up
    to now. UTF-8 is described in Annex R, published as Amendment 2.

[8] D. Goldsmith and M. Davis, ``Using Unicode with MIME'', RFC1641,
    July 1994

[9] F. Yergeau, ``UTF-8, a transformation format of Unicode and ISO
    10646'', RFC 2044, October 1996

[10] ITU-T Recommendation H.332 (1998): "Multimedia Terminal for
     Receiving Internet-based H.323 Conferences", ITU, Geneva.

[11] M. Handley, H. Schulzrinne, E. Scholler and J. Rosenberg ``SIP:
     Session Initiation Protocol'', RFC 2543, March 1999.

[12] H. Schulzrinne, A. Rao and R. Lanphier, ``Real Time Streaming
     Protocol (RTSP)'' RFC 2326, April 1998.

[13] S. Bradner, ``Key words for use in RFCs to Indicate Requirement
     Levels'', RFC 2119, March 1997.