

MTAS Interface to MRF (Mr)

INTERWORK DESCR

© Ericsson AB 2012

Disclaimer Statements

No part of this document may be reproduced in any form without the written permission of the copyright owner.

The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing.

Ericsson shall have no liability for any error or damages of any kind resulting from the use of this document.

Trademark List

All trademarks are the property of their respective owners.

Contents

1	Scope and Purpose	4
1.1	Interface Entities	4
1.2	Interface Role	5
1.3	Services	5
1.4	Encapsulation and Addressing	7
1.4.1	Mr Interface	7
1.4.2	Cr interface	8
2	Procedures	8
2.1	Overview	8
2.2	Lower Level Procedures	8
2.3	Announcement Service, not chained and not segmented	9
2.3.1	Playing a simple announcement	9
2.3.2	Playing simple announcement repeatedly forever	10
2.3.3	Suspending a repeatedly played simple announcement	11
2.3.4	Changing announcement parameters	13
2.4	Announcement Service, chained or segmented	14
2.4.1	Playing chained announcements	14
2.4.2	Playing segmented announcements	15
2.5	Conference Service	16
2.6	Prompt and Collect Service	18
2.6.1	Basic user input control	18
2.7	Configurable Announcement Parameters	21
3	Information Model	24
3.1	General	24
3.2	Announcement Service, not chained and not segmented	24
3.3	Announcement Service, chained or segmented	25
3.4	Conference Service	27
3.5	Prompt and Collect Service	28

4	Formal Syntax or Schema	28
5	Related Standards.....	28
6	Video support.....	28
7	Terminology	29
7.1	Abbreviations	29
7.2	Definitions	29
8	References.....	29

Document History

Rev	Date	Sign	Comment
A	2013-04-09	ealefaz	S-CSCF is used
A	2013-08-06	edmhov	SIP Resource-Priority header.
C	2014-08-29	ethzts	Updated chapter 7 References, to make the document platform-agnostic.
D	2014-11-18	ethzts	<value> tag removed from chapter 2.4.2 Playing segmented announcements
E	2015-06-22	ebarcsk	VXML 2.0 is used by MTAS instead of 2.1
F	2016-03-24	xmilmat	MTASv 1.0 -Updated References, CBA link to MOM
G	2016-05-27	ebarcsk	Updated chapter 1.3 with announcement definitions. Updated 2.3.2 and 2.3.3. Updated 2.3.4. The whole document is spell checked.
H	2016-07-20	Ebarcsk	Update <filled> tag with <var name="inputmode" expr="invocation\$.inputmode"/> <exit namelist="inputmode invocation"/>
J	2016-10-21	Eptehal	Changed vxml generation Documentation of video handling More info about Configurable Announcement parameters
K	2017-03-10	Eptehal, ECSAMLL	Changed vxml file naming for allowing MFRs to enable caching HV59342 – vxml related changes about play path replacement
L	2018-10-31	esrgari	Provided Variable format for Segmented Announcement in voicexml used by MTAS

1 Scope and Purpose

1.1 Interface Entities

Within the IMS architecture the interface that is used by MTAS to control media services in the Multimedia Resource Function (MRF) is the Mr.

This document describes how MTAS deploys this interface.

Any routers or proxies that may be installed between MTAS and the MRF are excluded from the scope of this document.

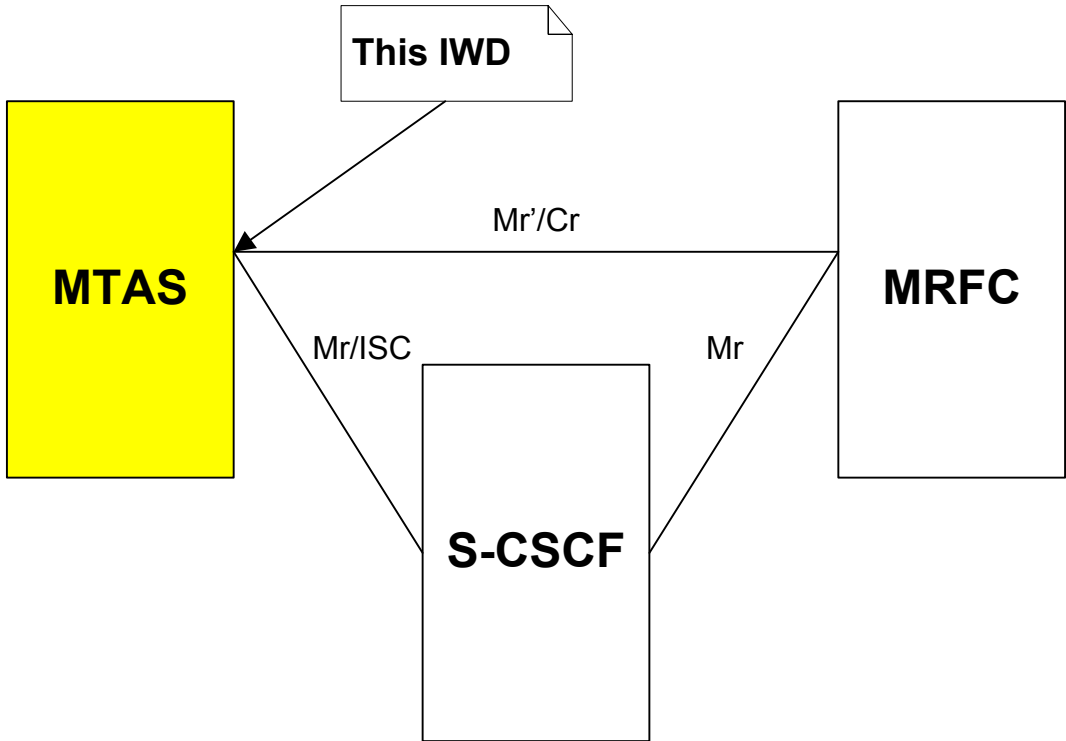


Figure 1 MTAS and MRF connectivity

Application Servers, like MTAS, can interact with the MRF either directly via the Mr' interface, or via S-CSCF using the ISC and the Mr interface.

The MRF will also interact with MTAS using the Cr interface to fetch VXML instructions. MTAS is using VXML version 2.0.

In the rest of the document Mr and Mr' interfaces are not distinguished, and referred to as Mr, any difference is highlighted explicitly.

1.2 Interface Role

In the context of the Mr interface, the role of MTAS is a client that uses network multimedia services.

1.3 Services

Table 1: Offered Services

Offered Service	Description
-----------------	-------------

- -

Table 2: Used Services

Used Service	Description
Announcement Service	Play an announcement without interruption and with no digit collection.
Conference Service	Create a conference for participants.
Prompt and Collect Service	Play an announcement and collect DTMF digits/voice answer, or play chained announcements.

The 3 types of announcement described in this document:

Simple Announcement:

A simple announcement is a video and/or audio message sent from the IMS network to the originating or terminating end-user. The content of the message is a single pre-defined media which does not change due to external conditions.

An example for the simple announcement is the announcement the caller (UE-A) receives when the called user (UE-B) is unreachable.

Chained Announcement:

Several simple announcements played one after another with the same media session.

Segmented Announcement:

It's a compound announcement that can consist of multiple parts called segments.

A segment type can be:

- *Provisioned:*
 - It refers to a pre-defined simple announcement available in MRF. Configured by the operator.
- *Standalone voice variable:*
 - Is an audio message with variable content.
 - The operator defines a name, a type of this variable, and if it is mandatory/optional.
 - The value of this variable is provided by MTAS.

An example of a segmented announcement is when the subscriber having a prepaid account gets notified about the current balance of his/her account. In this case, the provisioned part is the read out text "Your current balance is", and the standalone voice variable is - the read out actual balance of the account.

1.4 Encapsulation and Addressing

1.4.1 Mr Interface

The protocol on the Mr interface used by MTAS is NetAnn, as described in RFC4240 [1]. NetAnn is a protocol that describes how to use SIP to request basic network media services. Digit or voice invocation result is handled as described in chapter 4.2 SIP Mechanism in [2]. Voice Extended Markup Language 2.0 (VXML2.0) is used as described in [3]. On network layer either IPv4 or IPv6 can be used. In SIP headers and bodies IPv4 and IPv6 addresses can be mixed.

The services used on this interface are listed in Table 2: Used Services. No services are offered on this interface. When MTAS uses the Announcement Service, it expects that MRF is capable of suspending and resuming the announcement, and does not change its port, codec or IP address in subsequent SDP answers on the same SIP dialogue. The rationale behind this is that MTAS wants to avoid further SDP negotiations when it updates media attribute *recvonly/inactive* in a new SDP offer.

When MTAS uses the Announcement Service, it also expects that MRF is capable of changing the announcement. The rationale behind this is that MTAS wants to avoid further SDP negotiations when it updates streams and corresponding announcement URL.

When MTAS uses the Announcement Service and Conference Service, it expects that MRF does not reserve resources for streams where port is set to 0, as described in chapter 5.1, paragraph 2 in [4].

When MTAS uses the Conference Service, it expects that MRF does not change its port, codec or IP address in subsequent SDP answers on the same SIP dialogue. The rationale behind this is that MTAS wants to avoid further SDP negotiations when it updates port, codec or IP address in a new SDP offer.

When MTAS uses the Conference Service, it also expects that MRF reserves resources for streams with IP address set to 0.0.0.0 (or the content of attribute *MtasFunctionInvalidAddress* for IPv6), port set to a non-zero value, as described in chapter 8.4, last paragraph in [4]. The rationale behind this is that MTAS wants to reserve the resource prior to inviting a user.

When MTAS sends the URL pointing to a VXML file, MTAS itself will act as a policy server, and it will listen to HTTP requests on port 9080.

When sending a Mr session creating initial request over the Mr' interface, MTAS tries to contact the next element in the result of DNS SRV and/or A/AAAA lookup(s) when the request has encountered transport failure or timeout.

When sending INVITE messages over the Mr' interface in relation with a prioritized IMS session, MTAS includes Resource-priority header, a SIP extension defined in RFC4412 with the INVITE. MTAS uses *wps* and *ets* namespaces in Resource-priority header.

1.4.2 Cr interface

The Cr interface is used by the MRFC to fetch a VXML document that controls the behavior of the MRFC. The protocol used by the MRFC to fetch the VXML documents is HTTP. MTAS is using VXML 2.0.

2 Procedures

2.1 Overview

The requested service is specified in the URI with the service indicator and any appropriate parameters in the SIP INVITE sent to MRF on the Mr interface. The requested service indicators are:

Table 3 Requested Service Indicators

Service	Service Indicator
Announcement Service, not chained and not segmented	annc
Announcement Service chained or segmented	dialog
Conference Service	conf=<unique_id>
Prompt and Collect Service	dialog

2.2 Lower Level Procedures

N/A

2.3 Announcement Service, not chained and not segmented

2.3.1 Playing a simple announcement

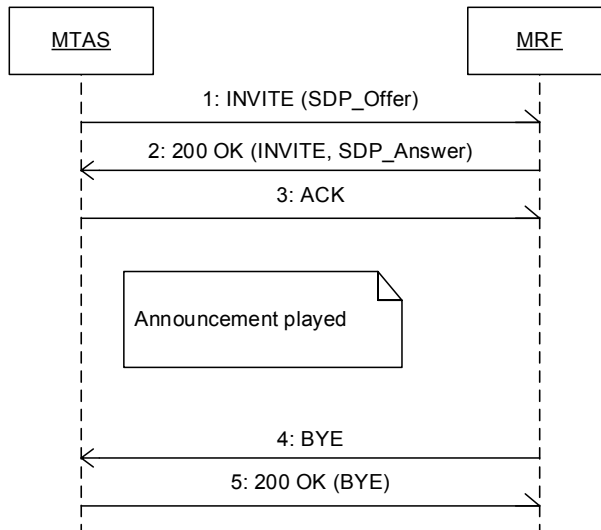


Figure 2 Playing a simple announcement

1. MTAS sends SIP INVITE to MRF with an SDP offer. URI is set to `sip:annc@<MRF_hostname>; play=<announcement_URL>` and can also contain `locale=<language_setting>`, `repeat=<nr_of_repetitions>`, `delay=<delay_time>` or `duration=<duration_time>`.

The announcement URL does not contain file extension, just a filename, like `file://opt/playcol/announcements/123`. MTAS expects that MRF is set up such a way, that it is a symbolic link pointing to a certain file.

2. MRF sends SIP 200 OK including an SDP answer.
3. MTAS sends a SIP ACK to MRF to trigger playing the announcement. MTAS expects that the announcement is only played after the ACK is received.
4. MRF sends a SIP BYE to MTAS when the announcement was played.
5. MTAS responds with a SIP 200 OK.

2.3.2 Playing simple announcement repeatedly forever

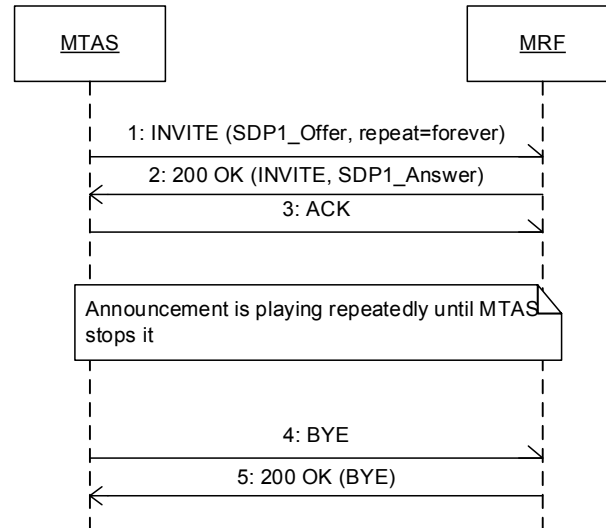


Figure 3 Playing simple announcement repeatedly forever

To check if announcement can be played repeatedly forever see: 2.7

1. MTAS sends SIP INVITE to MRF with an SDP offer. URI is set to sip:annc@<MRF_hostname>; play=<announcement_URL>; locale=<language_setting>; repeat=forever.

The announcement URL does not contain file extension, just a filename, like file://opt/playcol/announcements/123. MTAS expects that MRF is set up such a way, that it is a symbolic link pointing to a certain file.

2. MRF sends SIP 200 OK including an SDP answer.
3. MTAS sends a SIP ACK to MRF to trigger playing the announcement. MTAS expects that the announcement is only played after the ACK is received.
4. MTAS sends a SIP BYE to MRF to stop the announcement.
5. MRF responds with a SIP 200 OK.

2.3.3 Suspending a repeatedly played simple announcement

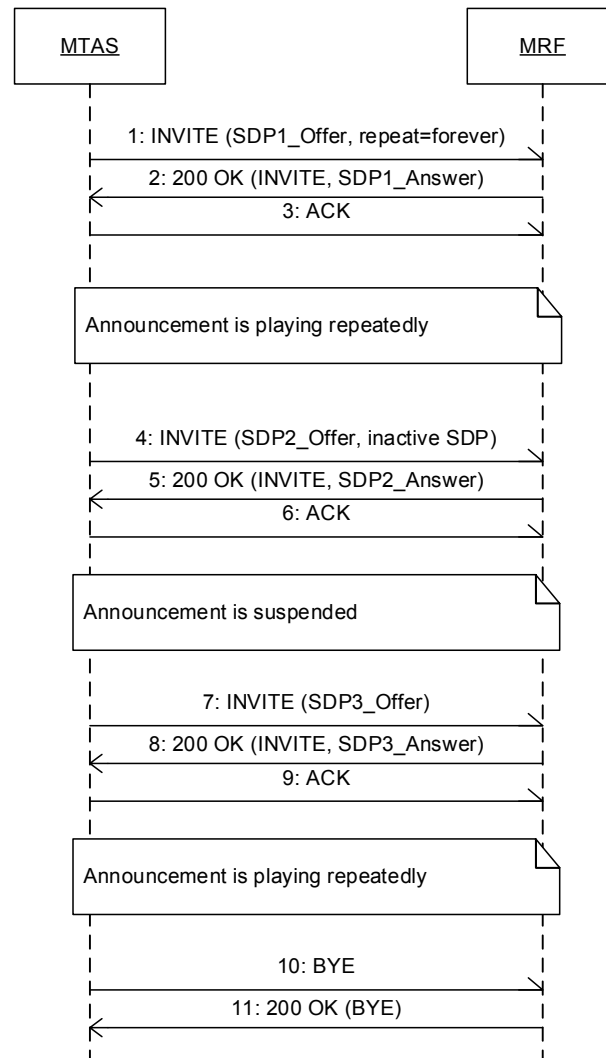


Figure 4- Suspending a repeatedly played simple announcement

1. MTAS sends SIP INVITE to MRF with an SDP offer. URI is set to sip:annc@<MRF_hostname>; play=<announcement_URL>; locale=<language_setting>; repeat=forever. The announcement URL does not contain file extension, just a filename, like file://opt/playcol/announcements/123. MTAS expects that MRF is set up such a way, that it is a symbolic link pointing to a certain file.

2. MRF sends SIP 200 OK including an SDP answer.

3. MTAS sends a SIP ACK to MRF to trigger playing the announcement. MTAS expects that the announcement is only played after the ACK is received.
4. MTAS sends a SIP re-INVITE to MRF with an inactive SDP offer in order to have the announcement suspended.
5. MRF responds with a SIP 200 OK including an SDP answer. MTAS expects that MRF does not change its IP address, port and codec compared to the SDP answer sent in step 2.
6. MTAS sends a SIP ACK to MRF.
7. MTAS sends a SIP re-INVITE to MRF with an SDP offer in order to have the announcement resumed.
8. MRF responds with a SIP 200 OK including an SDP answer. MTAS expects that MRF does not change its IP address, port and codec compared to the SDP answer sent in step 2.
9. MTAS sends an SIP ACK to MRF.
10. MTAS sends a SIP BYE to MRF to stop the announcement.
11. MRF responds with a SIP 200 OK.

2.3.4 Changing announcement parameters

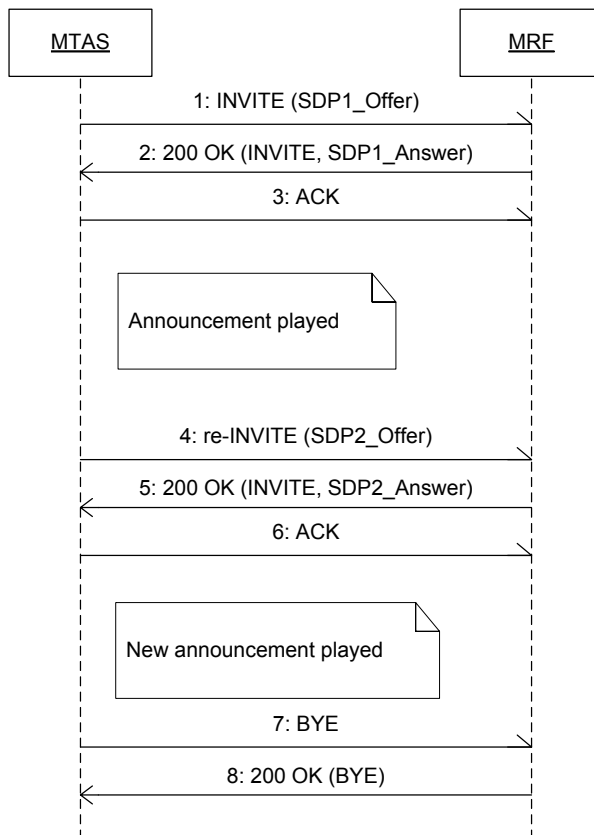


Figure 5 Changing announcement parameters

1. MTAS sends SIP INVITE to MRF with an SDP offer. URI is set to sip:annc@<MRF_hostname>; play=<announcement_URL>; locale=<language_setting>. The announcement URL does not contain file extension, just a filename, like file://opt/playcol/announcements/123. MTAS expects that MRF is set up such a way, that it is a symbolic link pointing to a certain file.
2. MRF sends SIP 200 OK including an SDP answer.
3. MTAS sends a SIP ACK to MRF to trigger playing the announcement. MTAS expects that the announcement is only played after the ACK is received.
4. MTAS sends a SIP re-INVITE to MRF with a new SDP offer and a new play URL. It can happen that the port or the codec is also changed in the new SDP offer.
5. MRF responds with a SIP 200 OK including an SDP answer.
6. MTAS sends a SIP ACK to MRF.
7. MTAS sends a SIP BYE to MRF to stop the announcement.

8. MRF responds with a SIP 200 OK.

2.4 Announcement Service, chained or segmented

2.4.1 Playing chained announcements

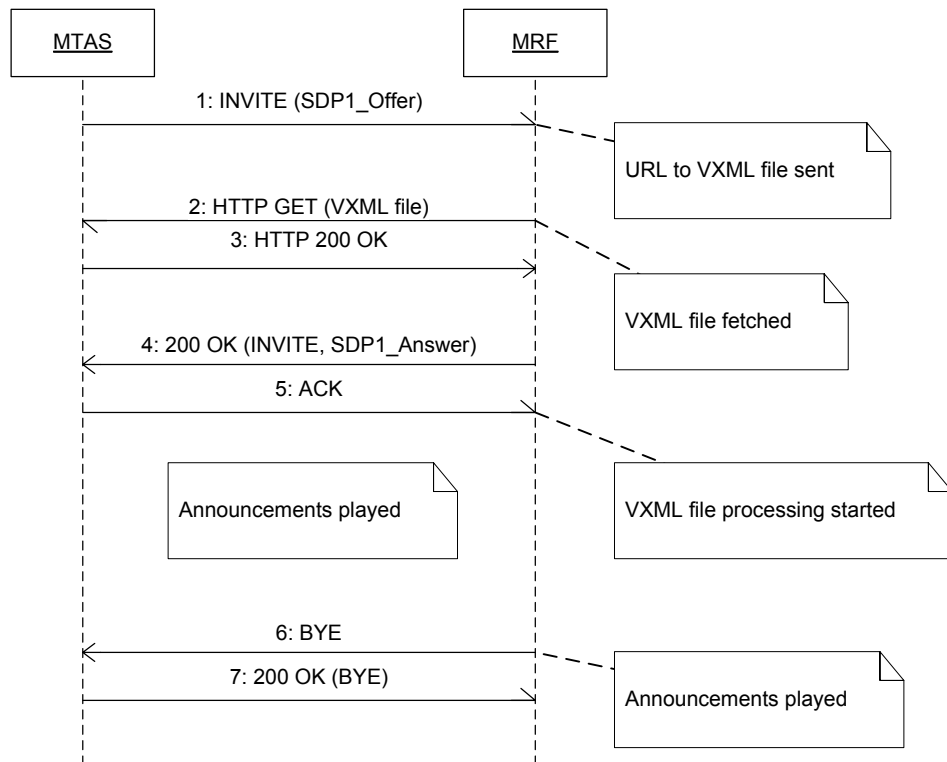


Figure 6 *Playing chained announcements*

1 MTAS sends a SIP INVITE to MRF with an SDP offer. URI is set to sip:dialog@<MRF_hostname>; voicexml=http://<policy_server_name>:<policy_server_port>/<vxml_file_location>, where vxml_file_location = NetAnnDialog/<user's call id>_<timestamp>.vxml

2 MRF sends a HTTP GET request to MTAS to the URL received in the INVITE.

3 MTAS returns the VXML file to MRF. The format of the file is as in the example:

```

<?xml version="1.0" encoding="utf-8"?>
<vxml version="2.0" xml:lang="<language-tag if provisioned>"
xmlns="http://www.w3.org/2001/vxml">
  <form>
    <block>

```

```

        <audio src=<announcement_1_URL>/>
        <audio src=<announcement_2_URL>/>
        <audio src=<announcement_3_URL>/>
    </block>
</form>
</vxml>

```

Note that in case language-tag is not provisioned, xml:lang attribute is missing.

4 MRF responds with SIP 200 OK (INVITE) to MTAS with the SDP answer.

5 MTAS requests MRF to process the VXML file by sending the SIP ACK.

The announcements pointed by the <announcement_1_URL>, <announcement_2_URL> and <announcement_3_URL> are played one after another.

6 When all announcements are played, MRF sends a SIP BYE to MTAS.

7 MTAS responds with a SIP 200 OK (BYE).

2.4.2 Playing segmented announcements

The service differs from the chained announcement in a sense of the content of the VXML document that is returned by MTAS at step 3.

```

<?xml version="1.0" encoding="utf-8"?>
<vxml version="2.0" xml:lang="<language-tag if provisioned>"
xmlns="http://www.w3.org/2001/vxml">
    <form>
        <block>
            <prompt>
                <audio src=<announcement_URL>/>
            </prompt>
            <prompt>
                <say-as interpret-as="<voice type>">
                    <voice value>
                </say-as>
            </prompt>
            <prompt>
                <say-as interpret-as="<voice type>">
                    <voice value>
                </say-as>
            </prompt>
        </block>
    </form>
</vxml>

```

The VXML document describes a vector of audio segments that appear as series of <prompt> child elements within a <block>. A segment can either be provisioned (<audio>) or standalone voice variable (<say-as>). A vector (<block>) can contain maximum 65 segments (<prompt>), and max. 32 <prompt> elements with <say-as> child elements.

The voice type is indicated in the 'interpret-as' attribute of the <say-as> element. The voice value is indicated in the content of the <say-as> element.

E.g: If voice type is "vxml:digits", the voice value is a series of decimal digits.

```
<say-as interpret-as="vxml:digits">
123456
</say-as>
```

2.5 Conference Service

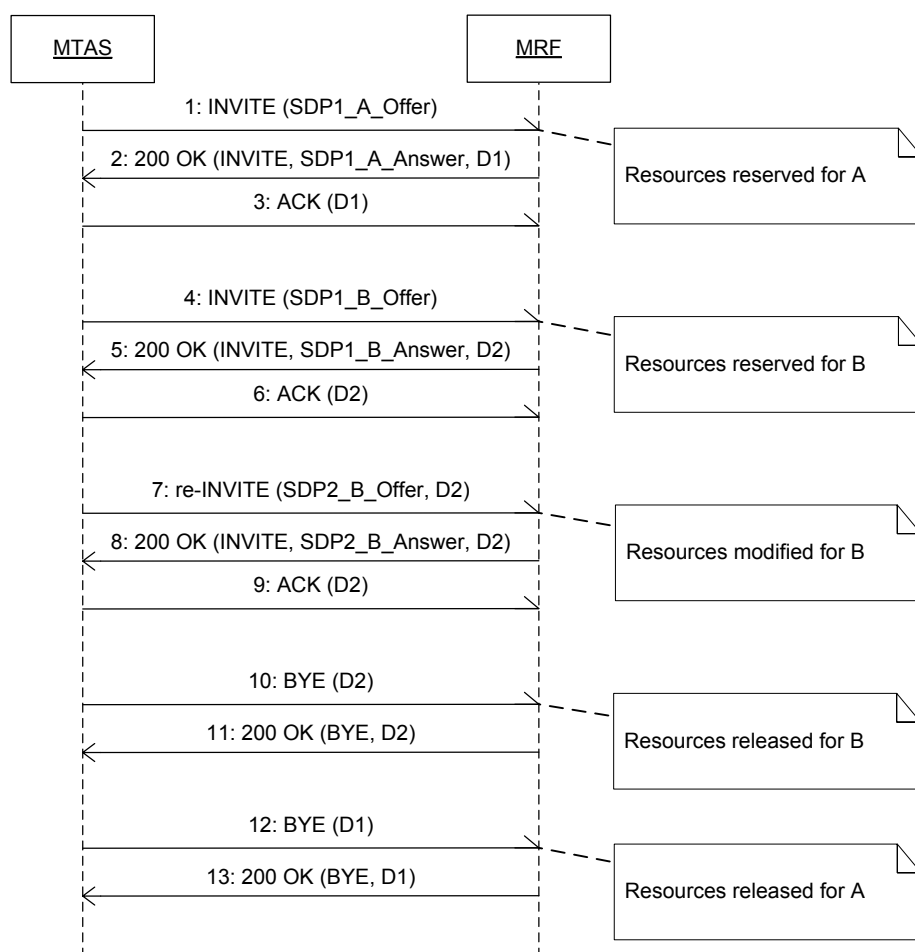


Figure 7 Conference Service

1. MTAS sends SIP INVITE to MRF with an SDP offer for user A. URI is set to sip:conf=<unique_id>@<MRF_hostname>.

2. MRF sends a SIP 200 OK with an SDP answer for user A. SIP dialog 1 is now established between MTAS and MRF.

3. MTAS sends SIP ACK to MRF on dialog 1.
 4. MTAS sends SIP INVITE to MRF with an SDP offer for user B with the following settings:
 - URI: There may be more MRFs in the network, and MTAS sends all INVITEs belonging to the same conference to the same MRF by setting the address in the URI to the same as the address in the Contact header field of the 200 OK received in step 2. Therefore the URI is set to sip:conf=<unique_id>@<MRF_address>. As MTAS wants to add B to the same conference as A, unique_id is set to the same value as in the first INVITE in step 1.
 - SDP: At this point MTAS has no information about B's IP address and port, so the IP address in the SDP is set to 0.0.0.0 (or the domain this.is.invalid for IPv6), port is set to a non-zero value.
 5. MRF sends a SIP 200 OK with an SDP answer for user B. SIP dialog 2 is now established between MTAS and MRF.
 6. MTAS sends SIP ACK to MRF on dialog 2.
 7. MTAS sends SIP re-INVITE to MRF with an updated SDP offer for user B on SIP dialog 2.
 8. MRF sends SIP 200 OK to MTAS with SDP answer for user B. MTAS expects that in the SDP answer the MRF does not change its port, codec or IP address compared to the SDP answer sent in step 5. The rationale behind this is that MTAS wants to avoid further SDP negotiations.
- More participants can be added the same way as in steps 4-9.
10. MTAS sends SIP BYE to MRF on SIP dialog 2 indicating that user B wishes to leave the conference.
 11. MRF sends SIP 200 OK to MTAS on SIP dialog 2. SIP dialog 2 is now closed.
 12. MTAS sends SIP BYE to MRF on SIP dialog 1 indicating that UE_A wishes to leave the conference.
 10. MRF sends SIP 200 OK to MTAS on SIP dialog 1. SIP dialog 1 is now closed.
- If further participants were added, steps 10-11 are repeated on the corresponding SIP dialogs.

2.6 Prompt and Collect Service

2.6.1 Basic user input control

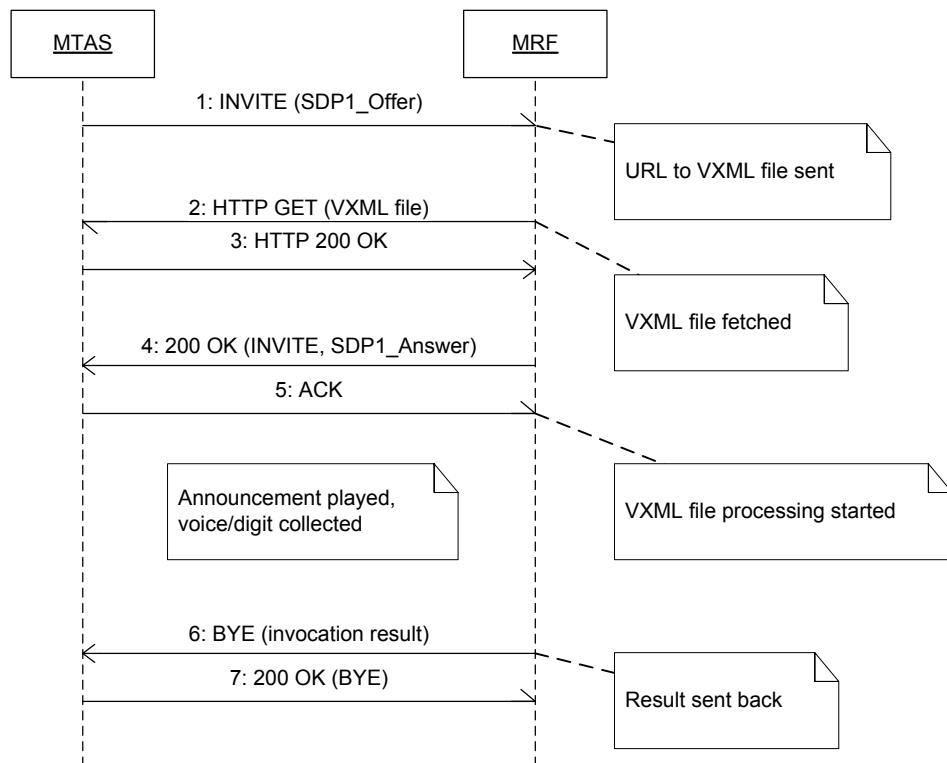


Figure 8 Collect voice/digit invocation

- 1 MTAS sends a SIP INVITE to MRF with an SDP offer. URI is set to sip:dialog@<MRF_hostname>; voicexml=http://<policy_server_name>:<policy_server_port>/<vxml_file_location>, where vxml_file_location = NetAnnDialog/<user's call id>_<timestamp>.vxml
- 2 MRF sends a HTTP GET request to MTAS to the URL received in the INVITE.
- 3 MTAS returns the VXML file to MRF.

The following example file has the setting listed below:

- It uses a start digit timeout of nine seconds. This is the allowed waiting time before the 1st button push, if exceeded 'no-input' event will be thrown.
- It uses an inter digit timeout of two seconds. This is the allowed waiting time between two button push:

- If exceeded when fewer digits entered, then the minimal length of the grammar then *'no-input'* event will be thrown.
- If exceeded when at least as many digits entered as the minimal length of the grammar, then input collection finishes and interpretation is continues with process <filled tag>.
- It has a predefined terminating character, if this entered digit collection is finished.
- Barging is defined true, this means announcement can be interrupted by user input.
- The initial announcement is initialAnnouncement.wav, by default this will not be played again. However, this can be played in *'no-input'* and *'no-match'* error cases (see later) if error handling is defined like that.
- It has a path for externally define ASR grammar for voice recognition.
- It collects one to two occurrences of the digits 0-9 and *, #. This condition is refined in <filled> in case of Service request specific set of pins. In this case the set is {1,22}
- At maximum 3 retrial is allowed for the user in case of unsuccessful pin collection.
- If no input is collected from the user noinput.wav will be played to inform user about the error. If maximum allowed retrial exceeded the document exits with an "noUserInput" error that is returned to MTAS in a SIP BYE.
- If input collected from the user does not match the grammar and the refined criteria in the <filled> tag noinput.wav will be played to inform user about the error. If maximum allowed retrial exceeded the document exits with an "noMatchRetryLimitExceeded" error that is returned to MTAS in a SIP BYE.

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml xmlns="http://www.w3.org/2001/vxml" version="2.0">
  <form id="promptandcollect">
    <field name="invocation">
      <property name="timeout" value="9s" />
      <property name="interdigittimeout" value="2s" />
      <property name="termchar" value="#" />
      <property name="bargein" value="true" />
    </field>
    <prompt>
      <audio src="initialAnnouncement.wav" />
    </prompt>
    <prompt count="2"/>
    <grammar src="asrgrammar.grxml" mode="voice"/>
    <grammar mode="dtmf" root="pin">
      <rule id="pin" scope="public">
        <one-of>
          <item repeat="1-2">
            <ruleref uri="#digit" />
          </item>
        </one-of>
      </rule>
    </grammar>
  </form>
</vxml>
```

```

        </one-of>
    </rule>
    <rule id="digit" scope="public">
        <one-of>
            <item>0</item>
            <item>1</item>
            <item>2</item>
            <item>3</item>
            <item>4</item>
            <item>5</item>
            <item>6</item>
            <item>7</item>
            <item>8</item>
            <item>9</item>
            <item>*</item>
            <item>#</item>
        </one-of>
    </rule>
</grammar>
<filled>
    <if cond="!( invocation == '1' || invocation == '22' )">
        <throw event="nomatch" />
    <else />
        <var name="inputmode" expr="'invocation$.inputmode'"/>
        <exit namelist="inputmode invocation" />
    </if>
</filled>
<noinput>
    <prompt>
        <audio src="file:/voices/noinput.wav" />
    </prompt>
    <var name="errorevent" expr="'noUserInput'"/>
    <exit namelist="errorevent" />
</noinput>
<nomatch>
    <prompt>
        <audio src="file:/voices/badinput.wav" />
    </prompt>
    <reprompt />
    <goto next="promptandcollect" />
</nomatch>
<nomatch count="3">
    <var name="errorevent" expr="'nomatchRetryLimitExceeded'"/>
    <exit namelist="errorevent" />
</nomatch>
</field>
<error>
    <var name="errorevent" expr="'_event'"/>
    <var name="errormessage" expr="'_message'"/>
    <exit namelist="errorevent errormessage" />
</error>
</form>
</vxml>

```

Note that in case language-tag is not provisioned, xml:lang attribute is missing.

- 4 MRF responds with SIP 200 OK (INVITE) to MTAS with the SDP answer.
- 5 MTAS requests MRF to process the VXML file by sending the SIP ACK.

The announcement pointed by the <announcement_URL> is played, then MRF collects voice and/or DTMF invocation using the grammars pointed by <asr_grammar_URL> and <dtmf_grammar_URL>.

6 MRF sends a SIP BYE to MTAS including the invocation result.

7 MTAS responds with a SIP 200 OK (BYE).

2.7 Configurable Announcement Parameters

MTAS enables the arbitrary configuration of the play, repeat, delay and duration parameters of the Request-URI in the initial INVITE request when the Announcement Service is used on the Mr interface.

The parameters specified by the MtasAnnouncementParameter Managed Objects (see [5]) are copied to the Request-URI without any modifications.

Applying play parameter is a little more complex. If mtasAnnouncementParameterPlay is set to a non-empty value (path, uri) then it will be used. In case mtasMrControllerVxmlPathReplacementForPlayParameter is also set to a non-empty path or uri then the path will be replaced in mtasAnnouncementParameterPlay with this (filename and extension is not touched). If mtasAnnouncementParameterPlay not present, the mtasExtMrfcBaseurl will be applied to announcement code. If mtasExtMrfcBaseurl is also empty, the announcement code without any path is used.

The most services in MTAS request finite announcement. When the operator configures a continuous announcement but the service logic expects a finite announcement the operator configured repeat and duration parameters will be overridden and the service logic configured values will be applied.

Table about which service request infinite or finite announcement:

Service	CM attributes	Infinite announcement requested
Network Announcement for unregistered user	mtasMmtAudioAnnounceCode	False
	mtasMmtVideoOnlyCode	False
	mtasMmtAVVideoCode	False
	mtasMmtAVAudioCode	False
Local Ringback	mtasMmtLocalRingbackAnnouncementName	False
Credit Notification	mtasChargingProfileNoCreditAnnouncementName	False
	mtasChargingProfileCreditLimitReachedAnnouncementName	False
	mtasChargingProfileLow CreditPreSessionAnnouncementName	False
	mtasChargingProfileVeryLow CreditPreSessionAnnouncementName	False
	mtasChargingProfileLow CreditMidSessionAnnouncementName	False
	mtasChargingProfileVeryLow CreditMidSessionAnnouncementName	False
CDIV	mtasCDivAudioAnnounceCode	False
	mtasCDivVideoAnnounceCode	False
	mtasCDivAVAudioAnnounceCode	False

	mtasCDivAVVideoAnnounceCode	False
HOLD	mtasHoldAudioCode	True
	mtasHoldVideoOnlyCode	True
	mtasHoldAVVideoCode	True
	mtasHoldAVAAudioCode	True
Communication Waiting	mtasCwAudioAnnounceCode	False
	mtasCwVideoAnnounceCode	False
	mtasCwAVVideoCode	False
	mtasCwAVAAudioCode	False
SSC	Plenty...	False
Communication Barring	mtasAcrAudioAnnounceCode	False
	mtasAcrVideoOnlyCode	False
	mtasAcrAVVideoCode	False
	mtasAcrAVAAudioCode	False
	mtaslcbAudioAnnounceCode	False
	mtaslcbVideoOnlyCode	False
	mtaslcbAVVideoCode	False
	mtaslcbAVAAudioCode	False
	mtasOcbAudioAnnounceCode	False
	mtasOcbVideoOnlyCode	False
	mtasOcbAVVideoCode	False
	mtasOcbAVAAudioCode	False
	mtasOcbBCatAnnouncementName	False
	mtasOcbOpBCatAnnouncementName	False
	mtasOcbLocalnessBCatAnnouncementName	False
	mtasApLdmpAnnouncementName	False
Network Announcement	mtasNaAnnAudioCode	False
	mtasNaAnnVideoOnlyCode	False
	mtasNaAnnAVAAudioCode	False
	mtasNaAnnAVVideoCode	False
	mtasNaAnnTAudioCode	False
	mtasNaAnnTVVideoOnlyCode	False
	mtasNaAnnTAVAAudioCode	False
	mtasNaAnnTAVVideoCode	False
	mtasNaRaAudioCode	False
Short Number Dialing	mtasSndAudioCode	False
	mtasSndVideoCode	False
	mtasSndAVAAudioCode	False
	mtasSndAVVideoCode	False
Flexible Communication Distribution	mtasFcdAudioAnnounceCode	True
	mtasFcdVideoAnnounceCode	True
	mtasFcdAVAAudioAnnounceCode	True
	mtasFcdAVVideoAnnounceCode	True
	mtasFcdNegAudioAnnounceCode	False
	mtasFcdNegVideoAnnounceCode	False
	mtasFcdNegAVAAudioAnnounceCode	False
	mtasFcdNegAVVideoAnnounceCode	False
Parlay-X 3PCC	mtasPx3pccSetupAudioAnnounceCode	False
	mtasPx3pccSetupVideoAnnounceCode	False
	mtasPx3pccSetupAVAAudioAnnounceCode	False
	mtasPx3pccSetupAVVideoAnnounceCode	False
	mtasPx3pccRingToneAudioAnnounceCode	True
	mtasPx3pccRingToneVideoAnnounceCode	True

	mtasPx3pccRingToneAVAudioAnnounceCode	True
	mtasPx3pccRingToneAVVideoAnnounceCode	True
	mtasPx3pccSetupFailureAudioAnnounceCode	False
	mtasPx3pccSetupFailureVideoAnnounceCode	False
	mtasPx3pccSetupFailureAVAudioAnnounceCode	False
	mtasPx3pccSetupFailureAVVideoAnnounceCode	False
Carrier Select	mtasCsLocalDisallowedAudioAnn	False
	mtasCsLocalDisallowedVideoAnn	False
	mtasCsLocalDisallowedAVAudioAnn	False
	mtasCsLocalDisallowedAVVideoAnn	False
	mtasCsDestDisallowedAudioAnn	False
	mtasCsDestDisallowedVideoAnn	False
	mtasCsDestDisallowedAVAudioAnn	False
	mtasCsDestDisallowedAVVideoAnn	False
Usecall Admission Control	mtasUCacOrigAudioAnn	False
	mtasUCacOrigVideoAnn	False
	mtasUCacOrigAVAudioAnn	False
	mtasUCacOrigAVVideoAnn	False
Group Call Admission Control	mtasGCacOrigAudioAnn	False
	mtasGCacOrigVideoAnn	False
	mtasGCacOrigAVAudioAnn	False
	mtasGCacOrigAVVideoAnn	False
Communication Completion	mtasCcRecallAudioAnn	False
	mtasCcRecallVideoAnn	False
	mtasCcRecallAVAudioAnn	False
	mtasCcRecallAVVideoAnn	False
	mtasCcRingToneAudioAnn	False
	mtasCcRingToneVideoAnn	False
	mtasCcRingToneAVAudioAnn	False
	mtasCcRingToneAVVideoAnn	False
	mtasCcInvokeSuccessAudioAnn	False
	mtasCcInvokeSuccessVideoAnn	False
	mtasCcInvokeSuccessAVAudioAnn	False
	mtasCcInvokeSuccessAVVideoAnn	False
	mtasCcInvokeFailAudioAnn	False
	mtasCcInvokeFailVideoAnn	False
	mtasCcInvokeFailAVAudioAnn	False
	mtasCcInvokeFailAVVideoAnn	False
	mtasCcBusyToneAudioAnn	False
	mtasCcBusyToneVideoAnn	False
	mtasCcBusyToneAVAudioAnn	False
	mtasCcBusyToneAVVideoAnn	False
	mtasCcCallFailedAudioAnn	False
	mtasCcCallFailedVideoAnn	False
	mtasCcCallFailedAVAudioAnn	False
	mtasCcCallFailedAVVideoAnn	False
	mtasCcRetainedBusyToneAudioAnn	False
	mtasCcRetainedBusyToneVideoAnn	False
	mtasCcRetainedBusyToneAVAudioAnn	False
	mtasCcRetainedBusyToneAVVideoAnn	False
Communication Completion Busy Subscriber	mtasCcbsInvokePromptAudioAnn	False
	mtasCcbsInvokePromptVideoAnn	False
	mtasCcbsInvokePromptAVAudioAnn	False

	mtasCcbsIvrInvPromptAVVideoAnn	False
	mtasCcbsIvrInvPromptAudioAnn	False
	mtasCcbsIvrInvPromptVideoAnn	False
	mtasCcbsIvrInvPromptAVAudioAnn	False
	mtasCcbsIvrInvPromptAVVideoAnn	False
Communication Completion No Reply	mtasCcnrIvrInvPromptAudioAnn	False
	mtasCcnrIvrInvPromptVideoAnn	False
	mtasCcnrIvrInvPromptAVAudioAnn	False
	mtasCcnrIvrInvPromptAVVideoAnn	False
	mtasCcnrIvrInvPromptAudioAnn	False
	mtasCcnrIvrInvPromptVideoAnn	False
	mtasCcnrIvrInvPromptAVAudioAnn	False
	mtasCcnrIvrInvPromptAVVideoAnn	False
Dialed Number Mapping	mtasDnmAnnRejectInvalidDialedNumberLength	False
	mtasDnmAnnRejectLocalFormatNbr	False
	mtasDnmAnnRejectShortCodeNbr	False
Carrier Select Rn	mtasCsRnInvalidCscAnnouncementName	False
Explicit Communication Transfer Termination	mtasEctTermAnnName	False

Continuous announcement can be configured by the operator by setting the repeat parameter to “forever” and by omitting the duration parameter.

Operator configuration			Service logic configuration	Result taken from
Repeat	Duration	Type		
integer	""	finite	finite	operator
"forever"	""	continuous	finite	service
integer	integer	finite	finite	operator
"forever"	integer	finite	finite	operator
integer	""	finite	continuous	operator
"forever"	""	continuous	continuous	operator
integer	integer	finite	continuous	operator
"forever"	integer	finite	continuous	operator

Precedence rules for announcement parameters

3 Information Model

3.1 General

This section describes the URI parameters that are used by MTAS.

3.2 Announcement Service, not chained and not segmented

MTAS uses the following parameters to set the URI for Announcement Service:

Table 4 URI Parameters for Announcement Service, not chained and not segmented

Parameter	P	Comment
play	M	MTAS always sets it. MTAS does not add

		file extension to the filename. MTAS expects that the MRF is set up such a way, that it is a symbolic link pointing to a certain file.
repeat	O	If sent by MTAS this parameter has the value "forever" or a value in the range 1 to 127..
delay	O	If sent by MTAS this parameter has the value 0 to 32767.
duration	O	If sent by MTAS this parameter has the value 0 to 32767.
locale	O	MTAS may set it.
param[n]	O	Not used by MTAS.
extension	O	Not used by MTAS.

In the following example MTAS orders MRF at hostname ms2.example.net to play an announcement located at http://audio.example.net/allcircuitbusy:

```
sip:annc@ms2.example.net;
play=http://audio.example.net/allcircuitsbusy
```

MTAS handles all SIP responses. The NetAnn related responses are listed in the following table:

Table 5 SIP Responses for Announcement Service, not chained and not segmented

SIP code and reason phrase	Reason
200 OK	The URI in the INVITE is correct, MRF is capable of executing the service.
400 Bad Request	The INVITE was sent without a "play=" parameter, or MRF cannot retrieve the announcement.
404 Not Found	The MRF cannot find the referenced URI.
488 Not Acceptable Here	MRF cannot perform the requested service or does not recognize the service indicator.
503 Service Unavailable	There are no free resources in the MRF to execute the service.

3.3 Announcement Service, chained or segmented

MTAS uses the following parameters to set the URI for Announcement Service, chained or segmented:

Table 6 URI Parameters for Announcement Service, chained or segmented

Parameter	P	Comment
voicexml	M	MTAS always sets it.
vxml-keyword	O	Not used by MTAS.

In the following example MTAS orders MRF at mediaserver.example.net to process a VXML file located at http://vxmlserver.example.net/cgi-bin/script.vxml:

```
sip:dialog@mediaserver.example.net;
voicexml=http://vxmlserver.example.net/cgi-bin/script.vxml
```

MTAS handles all SIP responses. The NetAnn related responses are listed in the following table:

Table 7 SIP Responses for Announcement Service, chained or segmented

SIP code and reason phrase	Reason
200 OK	The URI in the INVITE is correct, MRF is capable of executing the service.
400 Bad Request	The INVITE was sent without the a "voicexml =" parameter
404 Not Found	The MRF cannot find the referenced URI.

MTAS uses the following data types to represent the variable part in segmented announcements in voicexml:

Table 8: Variable format for Segmented Announcement in voicexml

Data Type	Data Sub Type	MTAS Usage
time	-	MTAS includes in segmented announcement to represent time in 24-hour format. For example, 14:47 is represented as... <say-as interpret-as="time">1447</say-as>
time	hms12	MTAS includes in segmented announcement to represent time in 12-hour format. For example, 2:47 PM is represented as... <say-as interpret-as="time" format="hms12">1447</say-as>
time	hms24	MTAS includes in segmented announcement to represent time in 24-hour format. For example, 14:47 is represented as... <say-as interpret-as="time" format="hms24">1447</say-as>
date	-	MTAS includes in segmented announcement to represent date. For example, 18-Oct-2018 is represented as... <say-as interpret-as="date">20181018</say-as>
date	mdy	MTAS includes in segmented announcement to represent date. For example, Oct-18-2018 is represented as... <say-as interpret-as="date" format="mdy">20181018</say-as>
date	dmy	MTAS includes in segmented announcement to represent date. For example, 18-Oct-2018 is represented as... <say-as interpret-as="date"

		format="dmy">20181018</say-as>
number	-	MTAS includes in segmented announcement to represent an integer. For example, to prompt user to enter a digit, like, "please press "1" to confirm", it is represented as... <say-as interpret-as="number">1</say-as>
cardinal	-	MTAS includes in segmented announcement to represent an integer of cardinal type. For example, to represent maximum number of attempts (5 in this example) to enter a valid digit pattern, it is represented as... <say-as interpret-as="number" format="cardinal">5</say-as>
ordinal	-	MTAS includes in segmented announcement to represent Day of the month. For example, 14th October is represented as... <say-as interpret-as="ordinal">14</say-as>
date	m	MTAS includes in segmented announcement to represent month. For example, October is represented as... <say-as interpret-as="date" format="m">10</say-as>
digits	-	MTAS includes in segmented announcement to represent phone number in digits. For example, phone number 46812345678 is represented as... <say-as interpret-as="digits">46812345678</say-as>

3.4 Conference Service

There are no parameters in the URI for Conference Service.

In the following example MTAS orders MRF at hostname mediaserver.example.net to create a conference identified by uniqueIdentifier, and add the user to it:

```
sip:conf=uniqueIdentifier@mediaserver.example.net
```

MTAS handles all SIP responses. The NetAnn related responses are listed in the following table:

Table 9 SIP Responses for Conference Service

SIP code and reason phrase	Reason
200 OK	The URI in the INVITE is correct, MRF is capable of executing the service.
404 Not Found	The MRF cannot find the referenced URI.
486 Busy Here	Maximum number of allowed participants reached.
488 Not Acceptable Here	MRF cannot perform the requested

	service or does not recognize the service indicator.
503 Service Unavailable	There are no free resources in the MRF to execute the service.

3.5 Prompt and Collect Service

See chapter 3.3.

4 Formal Syntax or Schema

The formal syntax of SIP URIs used to invoke services is described in [1].

5 Related Standards

The MTAS implementation of the Mr interface is based on [1].

6 Video support

Video support limited in simple announcements and not supported in chained, segmented and prompt and collect use cases.

The media type(s) used in announcement is calculated from two input sources. The served user's SDP offer and the announcement's configuration via ldap. The latter can be defined in a MtasGaAnn Moc or in Service's configuration attributes.

The table below describes the logic:

<u>SDP</u> <u>Operator conf.</u>	Audio	Video	Audio & Video
Audio	Audio	Empty	Audio
Video	Empty	Video	Video
Audio & Video	Audio	Video	Audio & Video

In NetAnn, "play" parameter can carry only one media resource. When both audio and video should be used, we cannot apply both. In this case always audio is applied into the message.

Currently no <video> tag is generated in vxml bodies.

7 Terminology

7.1 Abbreviations

CSCF	Call Session Control Function
MRF	Media Resource Function
URL	Uniform Resource Locator
VXML	Voice Extended Markup Language

7.2 Definitions

NetAnn	The protocol described in [1].
Finite announcement	An announcement that has a specific end either by a configured finite repetition number (repeat) and/or by a configured duration value.
Continuous announcement	An announcement that plays continuously without a specific end by setting the repetition number to the infinite value and by omitting the duration parameter

8 References

- [1] RFC4240 - Basic Network Media Services with SIP
- [2] RFC5552 - SIP Interface to VoiceXML Media Services
- [3] Voice Extensible Markup Language (VoiceXML)2.0, W3C Recommendation 16 March 2004
- [4] RFC3264 - An Offer/Answer Model with the Session Description Protocol (SDP)
- [5] TSP: MTAS Parameter Description, 1/190 84-AVA 901 09/n**
CBA: Managed Object Model MTAS 155 54-LZN 765 0163/n**
- [6] RFC4412 Communications Resource Priority in SIP

**See the Customer or Support library for the Application System in question to get the correct document version.