XML Description

snom
VoIP phones

snom 4S
SIP Media Server
Version 2.0

snom 4S Media Server Version 2.0 XML Description

1. Edition 2002

Table of Contents

Overview

The description of the behaviour of an account type is done with XML configuration documents. These documents do not follow any open standard and they are a pragmatic approach to make account definition effective and efficient.

The behaviour description follows the model of a state machine. The XML document describes which state exists, how the media server should work in a specific state and how the media server moves on to the next state.

XML documents may define variables and their default values. These variables may be overridden by user setup and by a concrete media connection. Some variables have a special meaning described in this document.

The document framework looks like this:

```
<?xml version =“1.0“ standalone=“yes“?>
<media_definition>
</media_definition>
```

In the media definition part, three tags are allowed: option, start and state. These tags are described in the next three chapters.

1

Top-Level Tags

2.1 Options

2.1.1 Definition

The option defines a default value for a variable that is present in the media type. An example would be <option name="number">1234567</option>.

Options names are case-insensitive. Option names beginning with "x-" are not displayed to the user and therefore cannot be changed.

2.1.2 Predefines Variables

Some variables have a predefined meaning.

"x-type" is the name of the behaviour description which is displayed to the user. If this tag is not present, the file name for the behaviour description without the suffix is taken.

If "register" is set, the media server registers an account with the provided url. If the url is incomplete, the media server tries to complete it using the default proxy name provided in the media server settings.

The "pass" option is used for answering authentication requests in a simple way. The username must be the same as the user name of the registration.

For more sophisticated authentication, each media type may define up to ten authentication sets consisting of "auth_realm1-10", "auth_user1-10" and "auth_pass1-10".

If "x-selectable" is set to false, the account cannot be selected by the user. This is helpful when certain accounts should not be visible to the user.

"x-name" is used to store the name of the account and must not be present in a media description.

"x-location" is used to store the location where the account information is stored and must also not be used in a media description. This variable will be written with the concrete value of the account.

If "x-msg_send_mwi" is set to true, the account will send a message waiting indication after a new message has been stored.

"msg_mwi_destination" designates the destination where the message waiting indication is sent.

"msg_num" is a read only variable that contains the number of messages stored in the account. "msg_num_new" contains the number of new messages, "msg_num_old" the number of stored messages.

"msg_new" is set to true, if the current message is a new message.

"msg_date" contains the date of the message.

"msg_number" contains the value of the From header when the message was recorded.

"msg_id" identifies the current message with an ID (number).

"msg_last_new", "msg_last_old", "msg_last" indicate if the current message is the last new message, the last saved message of the last message at all.

"gui_id" is a internal variable that is used to identify the media connection.

"lang" contains the language code, e.g. en for English.

"digit" is a read only variable that stores the last pressed DTMF digit in a media connection.

"record-length" defines the maximum size of a message in seconds.

"from" contains the short form representation of the From header that is connected. This is the username part of the URL in case that the hostname matches the default proxy of the media server. Otherwise, it is both username and hostname of the caller.

"to" is the same as from, but taken from the To-header.

"f_contact" contains the value of the From header field in the invitation.

"t_contact" contains the value of the To header field in the invitation.

"f_url" contains the url part of the f_contact.

"t_url" contains the url part of the t_contact.

"f_un" contains the user part of the f_url. "f_hn" contains the host part of the f_url. Same for "t_un" and "t_hn".

"start" contains the time when the call was started.

"account" contains the name of the account which has been called on the media server.

"owner" is set to true if f_un matches t_un, f_hn matches t_hn and f_port matched t_port.

If "x-ring" is set to true, the media server does not pick up the call and plays media with 183 Session Progress instead.

"x-lineid" stores the line ID for the registration of the account.

"hangup" defines the number of seconds, after the media server will actively terminate a call.

"answer_delay" defines the number of seconds before the media server answers a call.

2.1.3 URL parameters

Parameters which are provided in the url are copied into the variable set of a media connection.

For example, if the media connection receives a call for sip: errexl@domain.com?error=404, the variable error will be set to 404. This allows easy control of the media server via url parameters. Internal parameters can also be controlled this way, e.g. if someone calls sip: abc@domain.com?x-ring=true, the media server will send media as ring back.

2.2 Start

The start tag defines in which state the media should start operation.

If a condition tag is present, the media server evaluates the condition. If the value if the expression is not true, it ignores the start tag. How conditions are evaluated is described below.

There may be several start tags present. The media server checks from top to bottom which start tag can be executed. There must be at least one start tag without condition.

2

2.3 States

2.3.1 Name

The name tag describes the name of the state. The name for the state must be unique in the behaviour file.

2.3.2 Audio

There may be several audio tags present for a state. These tags describe the audio output in the current state.

If the tag "random" is set to true, the media player will start playing the given file at a random position (between 20 and 80 % of the file). This is helpful when music on hold is selected.

The audio may be of type file or of type text. If the audio type is text, the value of the tag describes the file location where the audio file can be found.

If the audio type is text, the media server will convert the given text into file representation. Normal letters are spelled as they are. Special commands are indicated by brackets [and]. The following commands are supported:

Time hh:mm: Read the text as hours and minutes.

Number x: x is interpreted as number and read out.

Date time: The date is read relative to the current date. If the date is within the same day, the day announcement is omitted. If the day is within the last week, the day is put before the time. Otherwise, the media server reads an absolute date.

2.3.3 Recording

The record tag starts recording the media. The filename is provided as content of the tag.

2.3.4 Events

Events describe the actions that should be performed when something happens. Events have names (listed below) and conditions. Conditions are the same as in the start description.

Events trigger actions that are described in chapter 5.

The following event names are available:

"record_full" fires when the maximum record time has been reached.

"conference_join" fires when somebody has joined the conference.

DTMF input is described by a one-character string or by a three-character string with the first digit, a dash and the last digit (e.g. "3-7"). "key" describes any DTMF input.

"audio_end" is fired when the state has no more audio to play. If the media player does not find such an event, it starts playing from beginning again.

"redirect_fail" is fired when a transfer failed.

2.3.5 Conference Mode

By default, the media server keeps accounts separate. If indicated by the tag name "conference", the following values are defined:

"listen" is typically used when someone joins the conference. The participant can listen to what is being said. If audio is defined for the account, the other participants hear that audio.

"participate" is used for full participation in the conference. The media is fully mixed with the other channels.

Actions

3.1 goto state

Goto merely goes to the state indicated.

3.2 set var val

Set sets a variable to the indicated value within the connection. That means, the account status is not accected by this operation. This is e.g. helpful for collecting digits.

3.3 save var val

This is like set, but the variable is permanently stores in the account database. This is e.g. useful for changing passwords.

3.4 hangup

This action terminates the current connection.

3.5 delete file

This action deletes the indicated file

3.6 move sec

This action moves the current audio position by the indicated amount in seconds.

3.7 redirect number

The redirect command redirects the current media connection to the indicated number. The account stays active until the redirection succeeded or failed. Typically, it makes sense to move to another state that play music on hold or ringback.

3.8 msg_create

This action creates a new message and sets the variables msg_id, msg_new, msg_date, msg_number, msg_num, msg_num_new, msg_num_old, msg_last_new, msg_last_old, msg_last. However, it does not start recording.

3.9 msg_next

This action moves to the next message. Msg_next_new moves to the next new message, msg_next_old to the next saved message. If there is no next message, the media server does not change its position and writes a log message.

3.10 msg_first

This action moves to the first message. Msg_first_new moves to the first new message, msg_first_old to the firstsaved message. If there is no first message, the media server does not change its position and writes a log message.

3.11 msg_delete

This action deletes the current message and moves to the next message.

3.12 msg_save

This action marks the current message as saved.

3

Variable Substitution

When variables are embedded in values, the media server replaces the text with the value of the variable.

4.1 Variable Substitution

When the media server find the pattern ${variable}, it replaces the term with the value of the indicated variable.

The same effect can be achieved with the $variable notation, however this works only as long as the variable consists only of characters between a-z (case insensitive).

4.2 Mathematical Expressions

The media server supports primitive arithmetic. This is helpful in cases when simple incrementing of subtraction is required.

The syntax for algebraic expressions is $[expression]. The expression may contain variables and other expressions.

The operations +, –, * and / are supported. + and – have a higher precedence as / and *.

4.3 Dollar-Sign

If you want to have a Dollar-Sign $, just put a $$ in the code.

4

Examples

5.1 Music on Hold

This simple example implements a music on hold server. The user may select his favourite melody with DTMF key.

```
<?xml version ="1.0" standalone="yes"?>
<media_definition>

<!-- global options for music on hold -->
<option name="x-type">Music on Hold</option>
<option name="register"></option>
<option name="pass"></option>
<start>music0</start>

<state name="music0">
  <audio type="file" random="true">${audio}/music/moh0.wav</audio>
  <event name="1"><cmd>goto music1</cmd></event>
  <event name="2"><cmd>goto music2</cmd></event>
  <event name="3"><cmd>goto music3</cmd></event>
  <event name="4"><cmd>goto music4</cmd></event>
  <event name="5"><cmd>goto music5</cmd></event>
  <event name="6"><cmd>goto music6</cmd></event>
  <event name="7"><cmd>goto music7</cmd></event>
  <event name="8"><cmd>goto music8</cmd></event>
  <event name="9"><cmd>goto music9</cmd></event>
</state>

<!-- other states not printed here -->


</media_definition>
```

5.2 Mailbox

This a bit more complex example show most of the media server mailbox support funtionality.

```
<?xml version ="1.0" standalone="yes"?>
<media_definition>

  <!-- global options for the mailbox -->
  <option name="x-type">Mailbox</option>
  <option name="x-conference">false</option>
  <option name="x-msg_send_mwi">true</option>
  <option name="answer_delay">15</option>
  <option name="identity"></option>
  <option name="record_length">120</option>
  <option name="register"></option> <!-- for registering this
account -->
  <option name="pass">0000</option> <!-- initial password (4 digits)
-->
  <option name="msg_mwi_destination"></option>

  <!-- find the start state -->
  <start condition="equal ${owner} true">welcome_main_menu</start>
  <start condition="equal ${identity} ${from}">welcome_main_menu</
start>
  <start condition="equal ${x-mailbox_mode} abs_stand">mailbox_
disabled_stand</start>
  <start condition="equal ${x-mailbox_mode} abs_name">mailbox_
disabled_name</start>
  <start condition="equal ${x-mailbox_mode} abs_pers">mailbox_
disabled_pers</start>
  <start condition="equal ${x-mailbox_mode} mb_pers">mailbox_
enabled_pers</start>
  <start condition="equal ${x-mailbox_mode} mb_name">mailbox_
enabled_name</start>
  <start>mailbox_enabled_stand</start> <!-- default -->

  <!-- Welcome to the voicemail system of 123. Leave your message
after the beep -->
  <state name="mailbox_enabled_stand">
    <!-- define the audio output: -->
    <audio type="file">${audio}/${lang}/mb_you_have_reached_the_
voicemail_system.wav</audio>
    <audio type="file">${audio}/${lang}/mb_leave_msg_after_tone_two_
minutes.wav</audio>
    <audio type="file">${audio}/${lang}/bi_beep.wav</audio>

    <!-- define the behavior: -->
    <event name="audio_end"><cmd>msg_create</cmd><cmd>goto record_
message</cmd></event>
    <event name="0-9"><cmd>set collect ${digit}</cmd><cmd>goto
```

```
collect_digits</cmd></event>
    <event name="#"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
    <event name="*"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
  </state>

  <!-- Name mailbox greeting. -->
  <state name="mailbox_enabled_name">
    <!-- define the audio output: -->
    <audio type="file">${audio}/${lang}/mb_this_is_the_mailbox_
of.wav</audio>
    <audio type="file">accounts/${account}/name.wav</audio>
    <audio type="file">${audio}/${lang}/mb_please_leave_msg_after_
tone.wav</audio>
    <audio type="file">${audio}/${lang}/bi_beep.wav</audio>

    <!-- define the behavior: -->
    <event name="audio_end"><cmd>msg_create</cmd><cmd>goto record_
message</cmd></event>
    <event name="0-9"><cmd>set collect ${digit}</cmd><cmd>goto
collect_digits</cmd></event>
    <event name="#"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
    <event name="*"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
  </state>

  <!-- Personal mailbox greeting. -->
  <state name="mailbox_enabled_pers">
    <!-- define the audio output: -->
    <audio type="file">accounts/${account}/personal_announce.wav</
audio>
    <audio type="file">${audio}/${lang}/bi_beep.wav</audio>

    <!-- define the behavior: -->
    <event name="audio_end"><cmd>msg_create</cmd><cmd>goto record_
message</cmd></event>
    <event name="0-9"><cmd>set collect ${digit}</cmd><cmd>goto
collect_digits</cmd></event>
    <event name="#"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
    <event name="*"><cmd>set collect</cmd><cmd>goto collect_digits</
cmd></event>
  </state>
```

```
  <!-- Standard absence greeting. -->
  <state name="mailbox_disabled_standard">
    <audio type="file">${audio}/${lang}/mb_mailbox_not_activated_no_
messages_will_be_saved.wav</audio>
  </state>

  <!-- Name mailbox greeting. -->
  <state name="mailbox_disabled_name">
    <audio type="file">${audio}/${lang}/mb_this_is_the_mailbox_
of.wav</audio>
    <audio type="file">accounts/${account}/name.wav</audio>
    <audio type="file">${audio}/${lang}/mb_mailbox_not_activated_no_
messages_will_be_saved.wav</audio>
  </state>

  <!-- Personal mailbox greeting. -->
  <state name="mailbox_disabled_name">
    <audio type="file">accounts/${account}/absence.wav</audio>
  </state>

  <!-- Record a new message: -->
  <state name="record_message">
    <record>accounts/${account}/msg${msg_id}.wav</record>
    <event name="record_full"><cmd>goto say_thanks</cmd></event>
  </state>

  <state name="say_thanks">
    <audio type="file">${audio}/${lang}/mb_thank_you_for_
message.wav</audio>
    <event name="audio_end"><cmd>hangup</cmd></event>
  </state>

  <!-- Say how many enw messages we have: -->
  <state name="new_messages_0">
       <audio type="file">${audio}/${lang}/mb_you_have_no_new_
messages.wav</audio>
       <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>
  <state name="new_messages_1">
       <audio type="file">${audio}/${lang}/mb_you_have_1_new_
message.wav</audio>
       <event name="audio_end"><cmd>msg_first_new</cmd><cmd>goto
read_messages</cmd></event>
  </state>
  <state name="new_messages_x">
       <audio type="file">${audio}/${lang}/mb_you_have.wav</audio>
```

```
      <audio type="text">[number ${msg_num_new}]</audio>
      <audio type="file">${audio}/${lang}/mb_new_messages.wav</
audio>
      <event name="audio_end"><cmd>msg_first_new</cmd><cmd>goto
read_messages</cmd></event>
  </state>

  <!-- Say how many messages we have (old and new): -->
  <state name="saved_messages_0">
      <audio type="file">${audio}/${lang}/mb_you_have.wav</audio>
      <audio type="file">${audio}/${lang}/mb_no.wav</audio>
      <audio type="file">${audio}/${lang}/mb_messages.wav</audio>
      <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>
  <state name="saved_messages_1">
      <audio type="file">${audio}/${lang}/mb_you_have.wav</audio>
      <audio type="text">[number ${msg_num}]</audio>
      <audio type="file">${audio}/${lang}/mb_message.wav</audio>
      <event name="audio_end"><cmd>msg_first</cmd><cmd>goto read_
messages</cmd></event>
  </state>
  <state name="saved_messages_x">
      <audio type="file">${audio}/${lang}/mb_you_have.wav</audio>
      <audio type="text">[number ${msg_num}]</audio>
      <audio type="file">${audio}/${lang}/mb_messages.wav</audio>
      <event name="audio_end"><cmd>msg_first</cmd><cmd>goto read_
messages</cmd></event>
  </state>


  <!-- Read out new messages: -->
  <state name="read_messages">
    <audio type="file">${audio}/${lang}/mb_received_on.wav</audio>
    <audio type="text">[date ${msg_date}]</audio>
    <audio type="file">${audio}/${lang}/mb_user.wav</audio>
    <audio type="text">${msg_number}</audio>
    <audio type="file">accounts/${account}/msg${msg_id}.wav</audio>
    <audio type="file">${audio}/${lang}/mb_end_of_message.wav</
audio>
    <audio type="file">${audio}/${lang}/mb_repeat4_delete7_
save9.wav</audio>

    <!-- user does not interact: -->
    <event name="audio_end" condition="equal ${msg_last} false">
      <cmd>msg_save</cmd>
      <cmd>msg_next</cmd>
```

```
  <cmd>goto read_messages</cmd>
</event>

<event name="audio_end" condition="equal ${msg_last} true">
  <cmd>msg_save</cmd>
  <cmd>goto main_menu</cmd>
</event>

<!-- user interaction: delete the message -->
<event name="7" condition="equal ${msg_last} false">
  <cmd>delete accounts/${account}/msg${msg_id}.wav</cmd>
  <cmd>msg_delete</cmd>
  <cmd>goto read_messages</cmd>
</event>

<event name="7" condition="equal ${msg_last} true">
  <cmd>delete accounts/${account}/msg${msg_id}.wav</cmd>
  <cmd>msg_delete</cmd>
  <cmd>goto main_menu</cmd>
</event>

<!-- user interaction: next message -->
<event name="9" condition="equal ${msg_last} false">
  <cmd>msg_save</cmd>
  <cmd>msg_next</cmd>
  <cmd>goto read_messages</cmd>
</event>

<event name="9" condition="equal ${msg_last} true">
  <cmd>msg_save</cmd>
  <cmd>goto main_menu</cmd>
</event>

<!-- user interaction: repeat the message -->
<event name="4">
  <cmd>msg_save</cmd>
  <cmd>goto read_messages</cmd>
</event>

<!-- user interaction: step back 10 sec -->
<event name="1">
  <cmd>move -10</cmd>
</event>

<!-- user interaction: go to main menu -->
<event name="#">
```

```
      <cmd>msg_save</cmd>
      <cmd>goto main_menu</cmd>
    </event>
    <event name="*">
      <cmd>msg_save</cmd>
      <cmd>goto main_menu</cmd>
    </event>

  </state>

  <!-- Collect digits for authentication: -->
  <state name="collect_digits">
    <event name="0-9" condition="equal ${collect}${digit} ${pass}">
      <cmd>goto welcome_main_menu</cmd>
    </event>
    <event name="0-9" condition="wrong ${collect}${digit} ${pass}">
      <cmd>goto wrong_code</cmd>
    </event>
    <event name="0-9" condition="too_short ${collect}${digit}
${pass}">
      <cmd>set collect ${collect}${digit}</cmd>
    </event>
  </state>

  <state name="wrong_code">
    <audio type="file">${audio}/${lang}/mb_your_pin_code_is_not_
correct.wav</audio>
    <event name="0-9">
      <cmd>set collect ${digit}</cmd>
      <cmd>goto collect_digits</cmd>
    </event>
       <event name="audio_end"><cmd>set collect</cmd><cmd>goto
collect_digits</cmd></event>
  </state>

  <!-- Before the main menu: -->
  <state name="welcome_main_menu">
    <audio type="file">${audio}/${lang}/mb_welcome_owner.wav</audio>
       <event name="audio_end" condition="equal ${msg_num_new}
0"><cmd>goto main_menu</cmd></event>
    <event name="audio_end" condition="equal ${msg_num_new}
1"><cmd>msg_first_new</cmd><cmd>goto new_messages_1</cmd></event>
    <event name="audio_end"><cmd>msg_first_new</cmd><cmd>goto new_
messages_x</cmd></event>
  </state>
```

```
  <!-- The main menu:
       1: listen to messages
       2: select mailbox mode
       3: record name
       4: record annoucement
       5: record absence message
       6: change pin
       -->
  <state name="main_menu">
    <audio type="file">${audio}/${lang}/mb_main_menu.wav</audio>
    <event name="1" condition="equal ${msg_num} 0"><cmd>goto saved_
messages_0</cmd></event>
    <event name="1" condition="equal ${msg_num} 1"><cmd>msg_first</
cmd><cmd>goto saved_messages_1</cmd></event>
    <event name="1"><cmd>msg_first</cmd><cmd>goto saved_messages_x</
cmd></event>
    <event name="2"><cmd>goto select_mailbox_mode</cmd></event>
    <event name="3"><cmd>goto record_name</cmd></event>
    <event name="4"><cmd>goto record_greeting</cmd></event>
    <event name="5"><cmd>goto record_absence</cmd></event>
    <event name="6"><cmd>goto change_passcode</cmd></event>
  </state>

  <!-- Enable or disable mailbox:
       1: mailbox standard nec dr30f 5.8  dvts-pcmcia/sony fddh
       2: mailbox name
       3: mailbox personal
       4: absence standard
       5: absence name
       6: absence personal
   -->
  <state name="select_mailbox_mode">
    <audio type="file">${audio}/${lang}/mb_select_mode.wav</audio>
    <event name="1"><cmd>save x-mailbox_mode mb_stand</cmd><cmd>goto
selected_mailbox_standard</cmd></event>
    <event name="2" condition="equal ${x-name_available}
true"><cmd>save x-mailbox_mode mb_name</cmd><cmd>goto selected_
mailbox_name</cmd></event>
    <event name="2" condition="unequal ${x-name_available}
true"><cmd>goto record_name</cmd></event>
    <event name="3" condition="equal ${x-greeting_available}
true"><cmd>save x-mailbox_mode mb_pers</cmd><cmd>goto selected_
mailbox_personal</cmd></event>
    <event name="6" condition="unequal ${x-greeting_available}
true"><cmd>goto record_greeting</cmd></event>
    <event name="4"><cmd>save x-mailbox_mode abs_stand</
```

```
cmd><cmd>goto selected_absence_standard</cmd></event>
    <event name="5" condition="equal ${x-name_available}
true"><cmd>save x-mailbox_mode abs_name</cmd><cmd>goto selected_
absence_name</cmd></event>
    <event name="5" condition="unequal ${x-name_available}
true"><cmd>goto record_name</cmd></event>
    <event name="6" condition="equal ${x-absence_available}
true"><cmd>save x-mailbox_mode abs_pers</cmd><cmd>goto selected_
absence_personal</cmd></event>
    <event name="6" condition="unequal ${x-absence_available}
true"><cmd>goto record_absence</cmd></event>
  </state>

  <state name="selected_mailbox_standard">
    <audio type="file">${audio}/${lang}/mb_selected_mailbox_
standard.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>

  <state name="selected_mailbox_name">
    <audio type="file">${audio}/${lang}/mb_selected_mailbox_
name.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>

  <state name="selected_mailbox_personal">
    <audio type="file">${audio}/${lang}/mb_selected_mailbox_
personal.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>

  <state name="selected_absence_standard">
    <audio type="file">${audio}/${lang}/mb_selected_absence_
standard.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>

  <state name="selected_absence_name">
    <audio type="file">${audio}/${lang}/mb_selected_absence_
name.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
```

```
  </state>

  <state name="selected_absence_personal">
    <audio type="file">${audio}/${lang}/mb_selected_absence_
personal.wav</audio>
    <event name="key"><cmd>goto main_menu</cmd></event>
    <event name="audio_end"><cmd>goto main_menu</cmd></event>
  </state>


  <!-- Change passcode: -->
  <state name="change_passcode">
    <audio type="file">${audio}/${lang}/mb_change_passcode.wav</
audio>
    <event name="#"><cmd>goto main_menu</cmd></event>
    <event name="*"><cmd>goto main_menu</cmd></event>
    <event name="0-9"><cmd>set pw ${digit}</cmd><cmd>goto change_
pw1</cmd></event>
  </state>

  <state name="change_pw1">
    <event name="0-9" condition="length ${pw} 3"><cmd>set pw
${pw}${digit}</cmd><cmd>goto say_passcode</cmd></event>
    <event name="0-9"><cmd>set pw ${pw}${digit}</cmd></event>
    <event name="#"><cmd>goto change_passcode</cmd></event>
    <event name="*"><cmd>goto change_passcode</cmd></event>
  </state>

  <state name="say_passcode">
    <audio type="file">${audio}/${lang}/mb_passcode_is.wav</audio>
    <audio type="text">$pw</audio>
    <audio type="file">${audio}/${lang}/mb_ack_passcode.wav</audio>
    <event name="#"><cmd>save pass ${pw}</cmd><cmd>goto main_menu</
cmd></event>
    <event name="*"><cmd>goto main_menu</cmd></event>
  </state>


  <!-- Recording the name: -->
  <state name="record_name">
    <audio type="file">${audio}/${lang}/mb_record_name.wav</audio>
    <event name="1"><cmd>goto record_name1</cmd></event>
    <event name="2"><cmd>goto play_name1</cmd></event>
    <event name="#" condition="equal ${recorded_name}
true"><cmd>save x-name_available true</cmd><cmd>goto main_menu</
cmd></event>
```

```
    <event name="*"><cmd>goto main_menu</cmd></event>
  </state>

  <state name="record_name1">
    <record>accounts/${account}/name.wav</record>
    <event name="#"><cmd>set recorded_name true</cmd><cmd>save x-
name_available true</cmd><cmd>goto record_name</cmd></event>
    <event name="1"><cmd>set recorded_name true</cmd><cmd>goto
record_name1</cmd></event>
    <event name="2"><cmd>set recorded_name true</cmd><cmd>goto play_
name1</cmd></event>
  </state>

  <state name="play_name1">
    <audio type="file">accounts/${account}/name.wav</audio>
    <audio type="text">   </audio> <!-- Little pause -->
    <event name="#"><cmd>save x-name_available true</cmd><cmd>goto
main_menu</cmd></event>
    <event name="1"><cmd>goto record_name1</cmd></event>
    <event name="2"><cmd>goto play_name1</cmd></event>
    <event name="audio_end"><cmd>goto record_name</cmd></event>
  </state>

  <!-- Recording the greeting: -->
  <state name="record_greeting">
    <audio type="file">${audio}/${lang}/mb_record_greeting.wav</
audio>
    <event name="1"><cmd>goto record_greeting1</cmd></event>
    <event name="2"><cmd>goto play_greeting1</cmd></event>
    <event name="#" condition="equal ${recorded_greeting}
true"><cmd>save x-greeting_available true</cmd><cmd>goto main_menu</
cmd></event>
  </state>

  <state name="record_greeting1">
    <record>accounts/${account}/greeting.wav</record>
    <event name="#"><cmd>set recorded_greeting true</cmd><cmd>save
x-greeting_available true</cmd><cmd>goto record_greeting</cmd></
event>
    <event name="1"><cmd>set recorded_greeting true</cmd><cmd>goto
record_greeting1</cmd></event>
    <event name="2"><cmd>set recorded_greeting true</cmd><cmd>goto
play_greeting1</cmd></event>
  </state>

  <state name="play_greeting1">
```

```
    <audio type="file">accounts/${account}/greeting.wav</audio>
    <audio type="text">   </audio> <!-- Little pause -->
    <event name="#"><cmd>save x-greeting_available true</
cmd><cmd>goto main_menu</cmd></event>
    <event name="1"><cmd>goto record_greeting1</cmd></event>
    <event name="2"><cmd>goto play_greeting1</cmd></event>
    <event name="audio_end"><cmd>goto record_greeting</cmd></event>
  </state>

  <!-- Recording the absence: -->
  <state name="record_absence">
    <audio type="file">${audio}/${lang}/mb_record_absence.wav</
audio>
    <event name="1"><cmd>goto record_absence1</cmd></event>
    <event name="2"><cmd>goto play_absence1</cmd></event>
    <event name="#" condition="equal ${recorded_absence}
true"><cmd>save x-absence_available true</cmd><cmd>goto main_menu</
cmd></event>
  </state>

  <state name="record_absence1">
    <record>accounts/${account}/absence.wav</record>
    <event name="#"><cmd>set recorded_absence true</cmd><cmd>save x-
absence_available true</cmd><cmd>goto record_absence</cmd></event>
    <event name="1"><cmd>set recorded_absence true</cmd><cmd>goto
record_absence1</cmd></event>
    <event name="2"><cmd>set recorded_absence true</cmd><cmd>goto
play_absence1</cmd></event>
  </state>

  <state name="play_absence1">
    <audio type="file">accounts/${account}/absence.wav</audio>
    <audio type="text">   </audio> <!-- Little pause -->
    <event name="#"><cmd>save x-absence_available true</
cmd><cmd>goto main_menu</cmd></event>
    <event name="1"><cmd>goto record_absence1</cmd></event>
    <event name="2"><cmd>goto play_absence1</cmd></event>
    <event name="audio_end"><cmd>goto record_absence</cmd></event>
  </state>

</media_definition>
```

5.3 Conference Server

This example implements a sipmle conference server with optional password authentication.

```
<?xml version ="1.0" standalone="yes"?>
<media_definition>

  <!-- global options for the conference -->
  <option name="x-type">Conference</option>
  <option name="register"></option> <!-- for registering this
account -->
  <option name="pass"></option> <!-- initial password (4 digits) -->
  <start condition="equal ${pass}${participants} 0">only_
participant</start>
  <start condition="equal ${pass}">enter_conference</start> <!-- no
password: start immediately -->
  <start>welcome</start>

  <!-- „Welcome to the conference server" -->
  <state name="welcome">
    <audio type="file">${audio}/${lang}/co_welcome_conference_
code.wav</audio>
    <event name="0-9"><cmd>set collect ${digit}</cmd><cmd>goto
collect_digits</cmd></event>
  </state>

  <state name="collect_digits">
    <event name="0-9" condition="equal ${collect}${digit}
${pass}"><cmd>goto enter_conference</cmd></event>
    <event name="0-9" condition="wrong ${collect}${digit}
${pass}"><cmd>goto wrong_code</cmd></event>
    <event name="0-9" condition="too_short ${collect}${digit}
${pass}"><cmd>set collect ${collect}${digit}</cmd></event>
    <event name="*"><cmd>set collect</cmd><cmd>goto welcome</cmd></
event>
    <event name="#"><cmd>goto wrong_code</cmd></event>
  </state>

  <state name="wrong_code">
    <audio type="file">${audio}/${lang}/mb_your_pin_code_is_not_
correct.wav</audio>
    <event name="0-9">
      <cmd>set collect ${digit}</cmd>
      <cmd>goto collect_digits</cmd>
    </event>
```

```
    <event name="audio_end"><cmd>set collect</cmd><cmd>goto collect_
digits</cmd></event>
  </state>

  <!-- conference can be „listen" or „participate" -->
  <state name="enter_conference" conference="listen">
    <audio type="file">${audio}/${lang}/co_somebody_joined_
conference.wav</audio>
    <event name="audio_end"><cmd>goto talk</cmd></event>
  </state>

  <!-- We are the only participant in the conference -->
  <state name="only_participant">
    <audio type="file">${audio}/${lang}/co_only_participant_in_
conference.wav</audio>
    <audio type="file">${audio}/music/moh0.wav</audio>
    <event name="conference_join"><cmd>goto talk</cmd></event>
  </state>

  <state name="talk" conference="participate">
  </state>

</media_definition>
```

5.4 Error Explanation

This error-explanation description uses the error number provided in the URL. If this parameter is not present, it uses a default error message.

```
<?xml version ="1.0" standalone="yes"?>
<media_definition>

  <!-- global options for the type -->
  <option name="x-type">Error Information</option>
  <option name="x-selectable">true</option>
  <option name="x-ring">true</option>
  <option name="hangup">60</option>
  <option name="register"></option> <!-- for registering this
account -->
  <option name="pass"></option>
  <option name="default_message">ei_number_unavailable.wav</option>
<!-- this is the default -->

  <start cond="equal ${error} 100">play_known_msg</start>
  <start cond="equal ${error} 180">play_known_msg</start>
```

```
  <start cond="equal ${error} 181">play_known_msg</start>
  <start cond="equal ${error} 182">play_known_msg</start>
  <start cond="equal ${error} 183">play_known_msg</start>
  <!-- other codes omitted -->
  <start cond="equal ${error} 604">play_known_msg</start>
  <start cond="equal ${error} 606">play_known_msg</start>
  <start>message</start>

  <state name="message">
    <audio type="file">${audio}/${lang}/ei_beep.wav</audio>
    <audio type="file">${audio}/${lang}/bi_space.wav</audio>
    <audio type="file">${audio}/${lang}/${default_message}</audio>
    <audio type="file">${audio}/${lang}/bi_space.wav</audio>
  </state>

  <state name="play_known_msg">
    <audio type="file">${audio}/${lang}/ei_beep.wav</audio>
    <audio type="file">${audio}/${lang}/bi_space.wav</audio>
    <audio type="file">${audio}/${lang}/ei_server_ret.wav</audio>
    <audio type="file">${audio}/${lang}/bi_space.wav</audio>
    <audio type="file">${audio}/${lang}/ei_${error}.wav</audio>
    <audio type="file">${audio}/${lang}/bi_space.wav</audio>
  </state>

</media_definition>
```

5

5

snom technology Aktiengesellschaft
Pascalstr. 10E, 10587 Berlin, Germany
Phone: +49 (30) 39833-0
mailto: info@snom.de
http: www.snom.de
sip: info@snomag.de

snom USA Representation
ABP International, Inc.
Crestside Dr.
Coppell, Texas 75019, USA
Phone: +1-972-831-0280
sip: usa@snomag.de
mailto: usa@snom.de

snom
VoIP-phones