A NOVEL MECHANISM FOR MEDIA RESOURCE CONTROL IN SIP MOBILE NETWORKS

Noël CRESPI, Youssef CHADLI,

Institut National des Telecommunications
9, rue Charles Fourier – 91011 EVRY Cedex FRANCE
Authors: N.Crespi, Y.Chadli
emails: {noel.crespi, youssef.chadli}@int-evry.fr

***Abstract* - In UMTS Session Initiation Protocol (SIP) multimedia architecture, the service logic is located in the Application Servers (AS). AS may need to control bearer resources, for playing tones and announcements, for conferencing or for transcoding purposes. These bearer resources are located in a media server: the Media Resource Function (MRF) or Media Server. The interface between the AS and the MRF is currently not fully specified and does not allow offering the most basic services. This paper proposes a novel mechanism for the AS to control the MRF and allow complex media interactions. This mechanism is based on the use of VoiceXML to describe media scripts, on XML to report the results, and SIP INVITE and INFO messages to transport the interaction related information between the Application Server and the Media Resource Function.**

***Key words*: SIP, IP CN Multimedia Subsystem, UMTS, VoiceXML.**

1. Introduction

The Session Initiation Protocol (SIP) [7] is an application-layer control (signalling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences. SIP is standardised by IETF [7].

The introduction of the IP Multimedia Core Network (CN) Subsystem (IMS) in UMTS Release 5 represents a fundamental change in mobile networks. The capabilities offered in the networks and in the terminals, the merging of voice and Internet leverage the inherent ability of this architecture to handle new services. In this architecture standardised by 3GPP, SIP is the key protocol.

In SIP based networks, the Application Server is an entity in which the service logic is executed. Examples of services include video point to point, web push, collaborative applications like screen sharing and co-browsing, Presence, Immediate Messaging, conferencing, click-to-call, simultaneous ringing. A broad range of applications needs to establish complex dialogues (for DTMF recognition, digit collection, text to speech and vocal synthesis), or to manipulate media stream (for conferencing and transcoding). The existing model in the standards to provide bearer resources is based on two functional entities: a Media Resource Function (MRF) or Media Server that handles the media resources, and an Application Server (AS) that handles the service logic and the call control. The AS controls the MRF with a SIP interface. In 3GPP architecture, the MRF is the entity that provides the media resources for the multimedia services.

RFC 2976 [8] has extended the SIP protocol by defining another SIP method called INFO. This method allows transporting encapsulated application level information within an established SIP session. This paper extends the use of the INFO method to media resource control without impacting RFC 2976.

VoiceXML [13] is an application of the eXtensible Markup Language (XML) [14] defined by the World Wide Web Consortium (W3C). It is a language that defines dialogs between humans and machines. These dialogs include synthesised speech, digitised audio, and recognition of spoken and DTMF input, recording of spoken input, telephony, and mixed-initiative conversations. Its major goal is to bring the advantages of web-based development and content delivery to interactive voice response applications.

2. State of the art

Within 3GPP

3GPP introduces in UMTS Release 5 within the IP Multimedia CN Subsystem the following entities:

- The Application Server (AS) is an entity in which the service logic is executed.
- The Media Resource Function (MRF) is an entity that provides bearer resources: mixing of incoming media streams (e.g. for multiple parties), media stream source (for multimedia announcements) and media stream processing (e.g. audio transcoding, media analysis).

The principles for media resource control described in TS23.228 [1] and in TS23.218 [2] are not complete and need further enhancements. The standardised mechanisms allows an AS to play an announcement for a mobile originated IP multimedia session with the AS (acting as back-to-back User Agent) performing third party call control with the MRF. However,

- the standardised mechanism does not specify how to play multiple tones or announcements with variable parts (dates, numbers), and does not provide a solution to repeat announcements like for Intelligent Network mechanisms defined in CAMEL phase 2 (TS 23.078 [3]).
- the standardised mechanism does not specify how to repeat or control the duration of an announcement.
- the standardised mechanism does not specify how the MRF can report user interaction results to the AS, e.g. DTMF. This type of script is however essential to most services, e.g. prepaid or voicemail.
- the standardised mechanism does not allow the AS to send to the MRF complex media scripts such as user identification by dialling DMTF digits.

In UMTS Release 6, there is a specific Work Task for Multimedia Conferencing, which is being specified in TS 24.147 [4] and TR 29.847 [5]. However, the AS-MRF interface is left outside of the scope of these 3GPP specifications and therefore a need for further definition of this interface remains.

Emerging solutions provided by manufacturers

As the AS-MRF interface is not fully specified, certain emerging products are based on integrated AS/MRF in a single physical entity. This does not however fulfil the operator requirement to have two distinct physical entities.

Alternatively there are emerging solutions based on HTTP protocol to transport VoiceXML script execution results between the AS and the MRF. This mechanism is however complex for the applications as it introduces a new interface between the AS and the MRF. The media interaction with the user is associated with a SIP session (established between the user agents and the MRF) and the interaction information is transported via HTTP transactions. There is therefore a need to correlate the SIP session and HTTP transactions: this brings in additional complexity to the application development.

Conclusion

The interface based on SIP between the AS and the MRF is not currently sufficient to provide most of the service requirements. SIP is essential to establish a media path between the User Agent and the MRF, but there is a need for an additional mechanism to allow the AS to control the MRF.

3. Proposed mechanism

This paper proposes a new mechanism for the control of the Media Server resources by the Application Server that allow complex media interaction with the user. This mechanism is based:

- on the use of VoiceXML scripts to describe vocal and DTMF interactions. A VoiceXML script can be used to describe a simple announcement to be played to the user as well as a complex interaction using advanced functionalities like vocal recognition and DTMF recognition. This is developed in subsection 3.1.
- on the use of the existing SIP interface to transport media control information with the SIP messages INVITE and INFO. As these messages belong to the same SIP session for which the script is executed, there is no need for an additional correlation mechanism like with HTTP transactions (cf. section 2). This simplifies the implementation of the service logic in the AS and the MRF. This is developed in subsection 3.2.
- on a XML description of the result encapsulated in a SIP INFO message sent by the MRF to the AS. This is developed in subsection 3.3.

We give an example in section 4 to illustrate this mechanism.

3.1 Use of VoiceXML to describe script commands sent by the AS

TS 23.218 [2] states that the INVITE sent to the MRF “will contain sufficient information to play the appropriate tone or announcement”; this needs to be further defined. We describe in this subsection the use of VoiceXML scripts.

The AS may either send a VoiceXML script to the MRF or point to an URL identifying a VoiceXML script present in the MRF. We identify below several scenarios:

1) The AS sends an URL pointing to the script:
   1.a) The URL of the script to execute is transmitted as a parameter of the *request URI* of a SIP message. This scenario implies that a new parameter of the request URI should be standardised.
   1.b) A SIP URI is associated to each VoiceXML script. There is therefore a one-to-one binding of the address in the *request URI* to a VoiceXML script.
   1.c) The URL of the script to execute is transmitted as a content in the body of a SIP message. In the SIP message, the MIME type of the content of the body is set to *"application/uri"*.
2) The AS sends the VoiceXML script encapsulated in a SIP message. In the SIP message, the MIME type of the content of the body is set to *"application/voicexml+xml"*. Note that media resources (e.g. announcement recordings) are not embedded in the message but are present in the MRF. This method should not be used for scripts of length exceeding 1300 bytes.

   The choice of an option among the four proposed is for further study; it may depend on the type of application, on the size of the VoiceXML script and on service deployment constraints. It does not impact the mechanism we propose in other subsections.

3.2 SIP messages used for media resource control

In this subsection we describe the SIP messages that shall be used to transport media resource command and results. Note that the proposed mechanism does not modify the SIP session establishment between the User Agent (e.g. SIP terminal) and the MRF.

3.2.1 At session establishment: INVITE request

A SIP INVITE is sent by the AS to the MRF to initiate a SIP session. The INVITE request should be used to transport an initial script command to the MRF. It may point to a VoiceXML script present in the MRF or contain an encapsulated VoiceXML script.

Responses and the ACK messages sent back to the AS shall not contain VoiceXML or XML elements.

The body of the invite message may contain a SDP description in addition to the VoiceXML script description or URI. Hence, the AS and the MRF/MS Shall understand the MIME type of "multipart/mixed" as defined in RFC2046 [4].

At the reception of an INVITE request for a script execution, the MRF/MS shall send an error response.

3.2.2 During an established session: INFO request

When a SIP session is established between an AS and a MRF, there may be script commands and responses sent between those two entities. In this paper we propose to use the SIP INFO request to transport these elements. As stated in RFC 2976 [8], the INFO request must be sent within an established SIP session and must not change the state of this session. Therefore, the INFO request is the most appropriate message to transport media resource control information when a SIP session is already established.

The use of the INFO method allows several exchanges between the AS and the MRF during an existing session. It uncouples the SIP session establishment and the control of the media resource within the session. This is particularly useful when several scripts are to be executed in a single SIP session.

The INFO method may be used:

- For posting script commands. The AS may ask for the execution of several scripts within the same established SIP session. The INFO method may be used to transport an initial script command (as an alternative to INVITE) or a subsequent script command.
- For posting results. For each script command, the MRF shall answer with an INFO request. Note: we do not recommend that the MRF sends a BYE message when the script is executed as the service logic in the AS may require further user interaction during the session using subsequent INFO messages. The MRF should therefore only act as a resource and should not take the initiative to tear down a session by sending a BYE message.

The INFO request shall be treated as follow:

- MRF response: at the reception of an INFO request for a script execution, the MRF shall send a *200* OK response if it is able to achieve the service. Otherwise (e.g., there is no available resource), it should send a *480 'temporarily Unavailable'* or a *503 'Service Unavailable'* response
- AS response: when the AS receives an INFO request containing the results related to the execution of a script, it should answer automatically with a *200* OK response.

3.3 Script execution results

3.3.1 Information returned by the MRF

Depending on the script, the MRF may return to the AS results of the execution or not. In any case, the MRF shall inform the AS indicating the treatment result: scripts execution ended normally or with errors. Note that this paper only describes final results, i.e. sent at the end of the script execution. The possibility to send to the AS intermediate results is for further study (cf. section 5).

In our proposal, when the script is executed, the MRF shall send an INFO request to the AS in order to inform it of the end of the script and possibly to transmit execution results e.g. DTMF dialled by the user.

In VoiceXML 2.0 [13], the execution of an application ends in the following cases:

1. The VoiceXML standard states that the treatment of the <*exit*> element depends entirely on the context. If this element is encountered at the MRF, it shall provoke the end of the execution. This end is considered as normal.
   The <*exit*> element can contain one of the two following attributes:
   - The *namelist* attribute. This attribute specifies the variable names to return. The default is to return no variables.
     If a list of variable names is specified by this attribute, the MRF shall send to the AS, as a result of the execution, the list of the values of these variables associated with their names.
     The results sent shall be in a string format. The values of the ECMAscript [15] variables shall be converted into string before being sent. The ECMAscript objects cannot be sent but the MRF may send the properties of the object (example: date.day, date.hour).
     Note: the audio recordings cannot be sent as a result in the INFO request. Moreover, the role of the AS is not to manage media resources. However, the MRF may store a media recording in a HTTP server and send the corresponding URL to the AS as a result.

     Example:

     ```
     < first-name, dupont >
     < second-name, paul >
     <message,
     http://www.MRF/messages/paul/33.wav
     >
     ```

   - The *Expr* attribute. This attribute corresponds to an ECMAScript expression evaluating the value of the result to return. A unique value is then returned.
     We propose that the MRF shall send this value associated with the name *'result* '.

     Example:

     ```
     < result, 43424465 >
     ```

     If the <exit> element does not return any value, the MRF shall inform the AS about the end of the service execution by specifying that it is a normal end (cf. subsection 3.3.2).

2. The interpreter reaches the end of the script document and no other document is specified to continue the execution. In this case, the end of the script execution is considered as normal.

   The MRF shall inform the AS about the end of the service execution by specifying that it is a normal end (cf. subsection 3.3.2).

3. An abnormal event occurs for which the treatment by default is to exit the session. Example: an error happened while loading a document. The end of the execution is considered as abnormal.

   The MRF shall then inform the AS of the abnormal execution abnormal: it sends the description of the event that states the cause of the VoiceXML session termination.

3.3.2 Use of XML for posting script execution results

The most appropriate description to report results to the AS is to use a XML format, which is already widely used in the 3GPP standards, e.g. for Presence service [6]. This section proposes an XML schema to send script results to the AS.

This information structured in a XML format is sent in the body of the INFO request by using the MIME type *"application/MRF-control+xml"*.

At the end of a script execution, the MRF shall send to the AS an INFO request to report the script execution result. It shall contain:

- An information indicating a normal execution (the interpreter has reached the end of the script or has encountered the *exit* element) or an error (an event provoked the exit of the execution).
- If the execution ends normally and the interpreter has encountered an <*exit*> element containing an *expr* or *namelist* attribute, this information shall be associated with a list of variable names and their values. If the *expr* attribute is present, the list shall contain only one element (result, value of *expr*).

The XML document sent by the MRF to the AS shall therefore contain an element <TerminationStatus> which can be associated with an element <Results>.

- The <TerminationStatus> element: contains a mandatory attribute called *status* that can take either the value "1" indicating a successful execution or the value “0” indicating an error in the treatment. The error cause is contained in the optional attribute: *event*.

- The <Results> element: contains a list of elements of the *result* type. The *result* element contains two mandatory attributes :
    - *nomVar* attribute*:* a string type that corresponds to the name of the *result* variable.
    - *valVar* attribute*: a* string type that corresponds to the value of the *result* variable.

3.3.3 Proposed XML schema

The namespace URI for elements defined in this paper is an URN conforming to RFC 2141 [9] and using the namespace identifier *'ietf'* defined by the RFC 2648 [10] and extended by [11]. This URN is: *“urn:ietf:params:xml :ns :MRF-control”*

```
<?xml version="1.0" encoding="UTF-8"?>
  <xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:ietf:params:xml:ns:MRF-control"
            xmlns="http://www.ietf.org"
            elementFormDefault="qualified">


< xsd :element name = “TerminationStatus “  >
  <xsd : complexType>
    <xsd:attribute name=”status” use=”required”>
     <xsd:simpleType >
       <xsd :restriction base = “xsd :integer”>
         <xsd:enumeration value = “0 ” />
         <xsd:enumeration value =  “1” />
       </xsd:restriction>
     <xsd:simpleType>
  <xsd:attribute name=  “event” use=”optional”>
         type=”xsd:string” />
  </xsd:complexType >
</xsd element>

<xsd:element name = “Results”  minOccurs =”0”>
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name = “result”>
           <xsd:complexType>
             <xsd:attribute name =” varName”
                type = “xsd:string”
                use   =”required” />
             <xsd:attribute name = “varValue”>
                type = “xsd:string”
                use   =”required” />
           </xsd:complexTytpe>
         </xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:elment>

</xsd:schema>
```

4. Example: simplified prepaid service

This section gives an example of simplified prepaid service using the mechanism described in section 3. A user refills his/her prepaid account by dialling a SIP or TEL URI service access; he/she is then prompted to enter a refill number. After verification of the number, an announcement is played to the user indicating the new value of his/her credit account.

Figure 1 gives a call flow for this service. Note that SIP proxies are not represented in this figure.

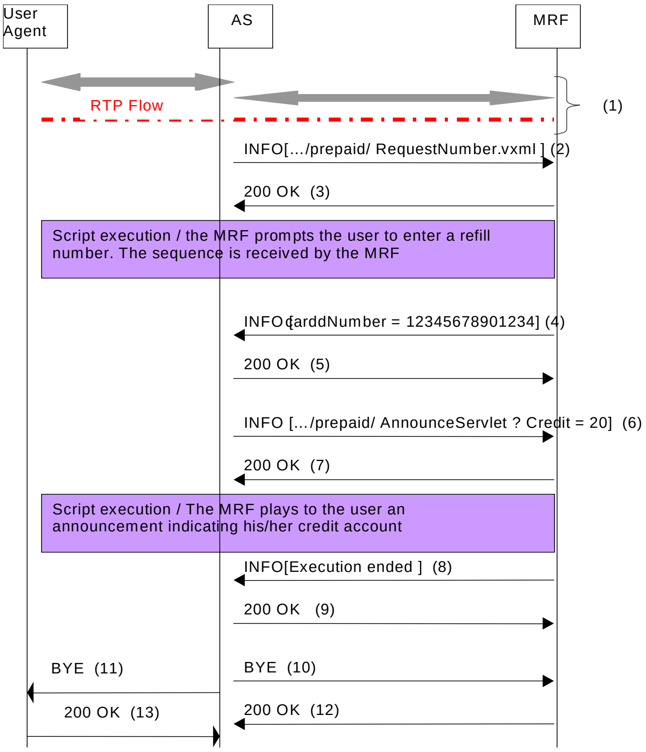
User Agent
AS
MRF
RTP Flow
(1)
INFO[.../prepaid/ RequestNumber.vxml] (2)
200 OK (3)
Script execution / the MRF prompts the user to enter a refill number. The sequence is received by the MRF
INFOc[arddNumber = 12345678901234] (4)
200 OK (5)
INFO [.../prepaid/ AnnounceServlet ? Credit = 20] (6)
200 OK (7)
Script execution / The MRF plays to the user an announcement indicating his/her credit account
INFO[Execution ended] (8)
200 OK (9)
BYE (11)
BYE (10)
200 OK (13)
200 OK (12)

Figure 1: simplified prepaid call flow (account refill)

(1) The AS plays the role of a third party call control. It creates a SIP session with the user and another one with the MRF. Moreover, it establishes the media flow directly between the user and the MRF. The SIP session establishment, which is described in TS 23.218 [2], is not shown in figure 1.

(2) The AS selects a script to request the user the enter a refill card number. It sends an INFO request to the MRF containing in its body the URL of the corresponding VoiceXML script (e.g.: …/prepaid/RequestNumber.vxml). The content-type header field of this message is set to *“application/uri”*.

Note that the script command may alternatively be sent in an INVITE request.

(3) The MRF responds with a *200* OK message because it is able to execute the script.

(4) At the end of the script execution, the MRF sends an INFO request containing in its body a XML script that encapsulates the refill card number provided by the user. The content-type header field of this request is set to *“application/ MRF-control+xml”*.

The XML script contained in the INFO request body is:

```
<?xml version=”1.0” encoding=”utf-8” ? >
<MRF-control>
   <TerminationStatus> Success  </TerminationStatus>
   <Results>
      <Result>
         varName = “cardNumber”
         varValue = “12345678901234”
      </Result>
   <Results>
</MRF-control>
```

(5) The AS responds with a *200* OK response.

(6) After checking the card number given by the user, the AS sends an INFO request containing in its body the URL of the VoiceXML script (…/prepaid/AnnounceServlet ? Credit = 20) that allows to inform the user of the remaining credit in the prepaid account. This URL contains a parameter called credit that indicates the value of the credit and points to a Java Servlet destined to create the VoiceXML script dynamically with this parameter. The content-type header field of this request is set to *“application/uri”*.

(7) The MRF responds with a *200* OK message because it is able to execute the script.

(8) and (9) At the end of the script execution, the MRF sends an INFO request to the AS to inform of the end of the treatment. The AS responds with a *200* OK response. The content-type header field of this request is set to *“application/ MRF-control+xml”*.

The XML script contained in the INFO request body is:

```
<?xml version=”1.0” encoding=”utf-8” ? >
<MRF-control>
    <TerminationStatus> Success  </TerminationStatus>
</M-control>
```

(10) to (13) The AS terminates the SIP session with the MRF and with the user. Session termination is described in TS 32.218 [2] and is not show in figure 1.

5. Conclusion

We proposed to fill an existing gap in the 3GPP standards for media resource control. The mechanism is based on a VoiceXML description of the script, an XML description of the results, and the SIP messages INVITE and INFO to transport media resource information. This proposal to enhance the interface between the Application Server and the Media Resource Function leverages new range of services and fulfils media control requirements expressed in TS 23.218 [2].

Using the INVITE and INFO messages to transport scripts commands and results between the AS and the MRF avoids the introduction of another interface, e.g. based on HTTP protocol. It facilitates the treatment in the AS because these messages belong to the same SIP session for which the VoiceXML script is executed.

Finally, using existing web standards (XML, VoiceXML) in this proposal facilitates the development of innovating services. It allows using the web development tools (like Java Servelts) on the MRF side that are already used in the service logic development on the Application Server side. Using the same tools in both entities is a key advantage in building enhanced services.

References

[1] 3GPP TS 23.228: " IP multimedia subsystem; Stage 2".

[2] 3GPP TS 23.218: “IP Multimedia (IM) session handling; IM call model; Stage 2”.

[3] 3GPP TS 23.078: “Customised Applications for Mobile network Enhanced Logic (CAMEL) Phase 3 - Stage 2”

[4] 3GPP TS 24.147: “Conferencing using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3”.

[5] 3GPP TR 29.847: “Conferencing based on SIP, SDP and other protocols; Functional models, information flows and protocol details”.

[6] 3GPP TS 24.141: “Presence service using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3”.

[7] RFC 3261 (June 2002), “SIP : Session Initiation Protocol”.

[8] RFC 2976 (October 2000), “The SIP INFO Method”.

[9] RFC 2141 (May 1997), “URN Syntax”.

[10] RFC 2648 (August 1999), “URN Namespace for IETF Documents”.

[11] RFC 3023 (January 2001), “XML Media Types”.

[12] M. Mealling ‘ The IETF XML registry ’ Internet draft, Internet Engineering Task Force, July 2002. Work in progress.

[13] VoiceXML 0.2 : http://www.w3.org/TR/VoiceXML20/

[14] XML, eXtensible Markup Language: www.w3.org/XML/

[15] Specification EcmaScript : http://www.ecma international.org/publications/standards/ECMA-262.HTM