

Groupe de travail Réseau

Request for Comments : 5552

Catégorie : Sur la voie de la normalisation

Traduction Claude Brière de L'Isle

D. Burke, Google

M. Scott, Genesys

mai 2009

Interface SIP aux services de support VoiceXML

Statut du présent mémoire

Le présent document spécifie un protocole de l'Internet sur la voie de la normalisation pour la communauté de l'Internet, et appelle à des discussions et suggestions pour son amélioration. Prière de se référer à l'édition en cours des "Protocoles officiels de l'Internet" (STD 1) pour voir l'état de normalisation et le statut de ce protocole. La distribution du présent mémoire n'est soumise à aucune restriction.

Notice de droits de reproduction

Copyright (c) 2009 IETF Trust et les personnes identifiées comme auteurs du document. Tous droits réservés.

Le présent document est soumis au BCP 78 et aux dispositions légales de l'IETF Trust qui se rapportent aux documents de l'IETF (<http://trustee.ietf.org/license-info>) en vigueur à la date de publication de ce document. Prière de revoir ces documents avec attention, car ils décrivent vos droits et obligations par rapport à ce document.

Résumé

Le présent document décrit une interface SIP aux services de support VoiceXML. Couramment, les serveurs d'application qui contrôlent les serveurs de supports utilisent ce protocole pour de pures capacités de traitement VoiceXML. Ce protocole est un ajout au protocole MEDIACTRL et au mécanisme de paquetage.

Table des matières

1. Introduction.....	2
1.1 Cas d'utilisation.....	2
1.2 Terminologie.....	4
2. Établissement et terminaison de session VoiceXML.....	4
2.1 Identification de service.....	4
2.2 Initiation d'une session VoiceXML.....	6
2.3 Préparation d'une session VoiceXML.....	7
2.4 Transpositions de variable de session.....	7
2.5 Terminaison d'une session VoiceXML.....	8
2.6 Exemples.....	9
3. Prise en charge des supports.....	11
3.1 Offre/réponse.....	11
3.2 Support précoce.....	11
3.3 Modification de la session de supports.....	12
3.4 Codecs audio et vidéo.....	12
3.5 DTMF.....	13
4. Retour de données au serveur d'application.....	13
4.1 Mécanisme HTTP.....	13
4.2 Mécanisme SIP.....	13
5. Appel sortant.....	14
6. Transfert d'appel.....	15
6.1 Blind.....	15
6.2 Bridge.....	16
6.3 Consultation.....	17
7. Contributeurs.....	18
8. Remerciements.....	18
9. Considérations sur la sécurité.....	18
10. Considérations relatives à l'IANA.....	18
11. Références.....	19
11.1 Références normatives.....	19
11.2 Références pour information.....	20
Appendice A. Notes sur les références normatives.....	21

Adresse des auteurs.....	21
--------------------------	----

1. Introduction

VoiceXML [VXML20], [VXML21] est une norme du World Wide Web Consortium (W3C) pour créer des dialogues audio et vidéo qui présentent de la parole synthétisée, de l'audio numérisé, la reconnaissance d'entrée de clés parlées et multifréquence bitonalité (DTMF, *dual tone multi-frequency*) l'enregistrement de conversations audio et vidéo, de téléphonie et d'initiative mixte. VoiceXML permet des développements fondés sur la Toile et des paradigmes de livraison de contenu à utiliser avec des applications interactives de vidéo et de réponse vocale,.

Le présent document décrit une interface SIP [RFC3261] aux services de supports VoiceXML. Couramment, les serveurs d'application qui contrôlent des serveurs de supports utilisent ce protocole pour des pures capacités de traitement VoiceXML. SIP est chargé d'initier une session de supports au serveur de supports VoiceXML et de déclencher simultanément l'exécution d'une application VoiceXML spécifiée. Ce protocole est un ajout au protocole MEDIACTRL complet et au mécanisme de mise en paquet.

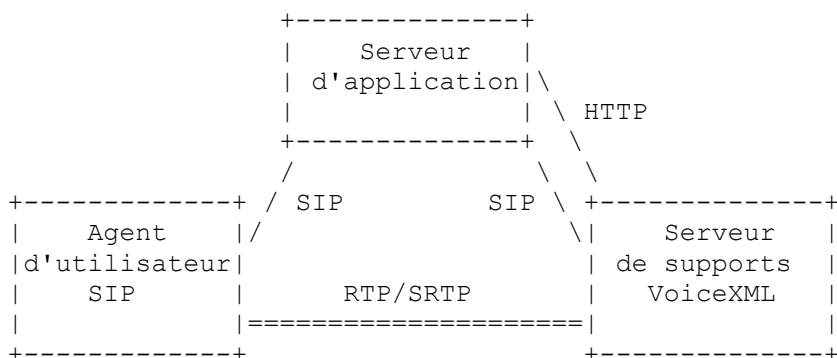
L'interface décrite ici développe un mécanisme pour identifier des services de support de dialogue d'abord décrits dans la [RFC4240]. L'interface a été mise à jour et étendue pour prendre en charge la Recommandation W3C pour VoiceXML 2.0 [VXML20] et VoiceXML 2.1 [VXML21]. Un ensemble de fonctions et extensions couramment mises en œuvre ont été spécifiées incluant la préparation de dialogue VoiceXML, d'appels sortants, la prise en charge de supports vidéo, et de transferts. Des transpositions de session VoiceXML variables ont été définies pour SIP avec un mécanisme extensible pour passer des valeurs spécifiques d'application dans l'application VoiceXML. Les mécanismes pour retourner les données au serveur d'application ont aussi été ajoutées.

1.1 Cas d'utilisation

L'utilisateur du service de support VoiceXML est dans le présent document appelé d'une façon générique un serveur d'application. En pratique, il est prévu que l'interface définie par le présent document soit applicable sur une large gamme de cas d'utilisation. Plusieurs cas d'utilisation prévus sont décrits ci-dessous.

1.1.1 Services IVR avec serveurs d'application

Les serveurs d'application SIP fournissent des services aux utilisateurs du réseau. Normalement, il peut y avoir plusieurs serveurs d'application dans le même réseau, chacun spécialisé dans la fourniture d'un service particulier. Dans la présente spécification et sans perte de généralité, on postule la présence d'un serveur d'application spécialisé dans la fourniture de services de réponse vocale interactive (IVR, *Interactive Voice Response*). Une configuration typique de ce cas d'utilisation est illustrée ci-dessous.



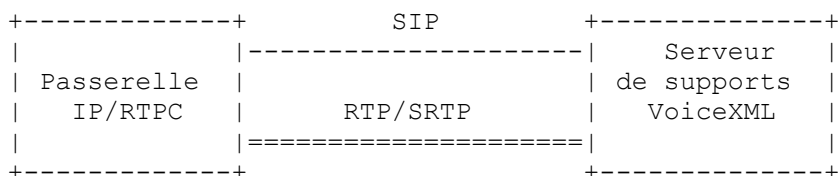
En supposant que le serveur d'application prend aussi en charge HTTP, l'application VoiceXML peut être hébergée sur lui et desservie via HTTP [RFC2616]. Noter cependant que le modèle de la Toile permet que l'application VoiceXML soit hébergée sur un serveur d'application séparé (HTTP) du serveur d'application (SIP) qui interagit avec le serveur de supports VoiceXML via la présente spécification. Il est aussi possible à une application VoiceXML statique d'être mémorisée en local sur le serveur de supports VoiceXML, déployant le mécanisme <data> de VoiceXML 2.1 [VXML21] et d'interagir avec un serveur Web/d'application quand un comportement dynamique est requis. La viabilité des applications VoiceXML statiques est encore améliorée par les mécanismes définis au paragraphe 2.4, par lesquels le serveur d'application peut

rendre des informations spécifiques de session disponibles dans le contexte de session VoiceXML.

L'approche décrite dans le présent document est parfois appelée le "modèle de délégation" -- le serveur d'application délègue essentiellement le contrôle programmatique des interactions homme-machine à un ou plusieurs documents VoiceXML fonctionnant sur le serveur de supports VoiceXML. Durant les interactions homme-machine, le serveur d'application reste dans le chemin de signalisation et peut répondre aux résultats retournés du serveur de supports VoiceXML ou à d'autres événements de réseau externes.

1.1.2 Nœud de service IVR RTPC

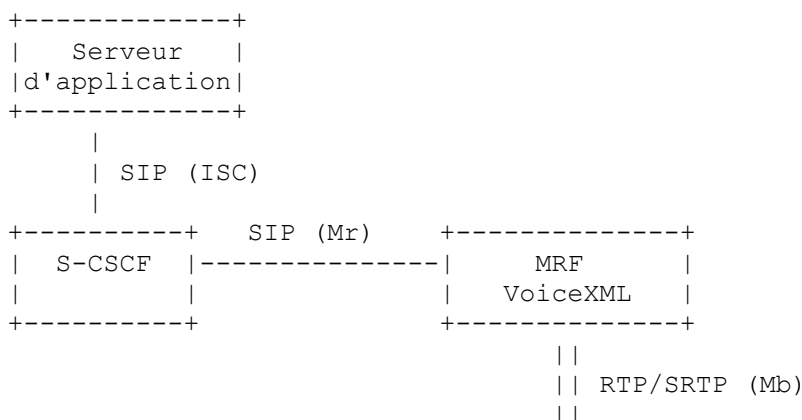
Bien que le présent document soit destiné à permettre une utilisation améliorée de VoiceXML comme composant de systèmes et services plus larges, il est prévu que les appareils qui ignorent complètement la présente spécification restent capables d'invoquer les services VoiceXML offerts par un serveur de supports VoiceXML conforme au présent document. Une configuration typique de ce cas d'utilisation est comme suit :



Noter aussi qu'au delà de l'invocation et de la terminaison d'un dialogue VoiceXML, la sémantique définie pour les transferts d'appels utilisant REFER est destinée à être compatible avec les passerelles standard existantes IP/RTPC (Réseau Téléphonique Public Commuté).

1.1.3 Fonction de ressource de support (MRF) IMS 3GPP

Le sous système multimédia IP (IMS, *IP Multimedia Subsystem*) du projet en partenariat de troisième génération (3GPP, *3rd Generation Partnership Project*) [TS23002] définit une fonction de ressource de support (MRF, *Media Resource Function*) utilisée pour offrir des services de traitement des supports comme la conférence, le transcodage, et l'invite/collecte. Les capacités offertes par VoiceXML sont idéales pour offrir de plus riches services de traitement des supports dans le contexte de la MRF. Dans cette architecture, l'interface définie ici correspond à l'interface "Mr" au contrôleur de MRF (MRFC, *MRF Controller*) ; la mise en œuvre de cette interface pourrait utiliser des éléments MRFC et MRFP (MRF Processeur) séparés (conformément à l'architecture IMS) ou pourrait être une MRF intégrée (comme dans la pratique courante).



Le diagramme ci-dessus est très simplifié et montre un sous ensemble des nœuds normalement impliqués dans les interactions de MRF. On devrait noter qu'alors que la MRF va principalement être utilisée par le serveur d'application via la fonction de contrôle de session d'appel servante (S-CSCF, *Serving Call Session Control Function*) il est aussi possible que les appels soient acheminés directement à la MRF sans impliquer de serveur d'application.

Bien que ceci soit décrit dans les termes de l'architecture du 3GPP IMS, il est prévu que cela soit aussi applicable aux architectures 3GPP2, de réseau de prochaine génération (HGH, *Next Generation Network*), et de PacketCable qui sont convergentes avec les normes 3GPP IMS.

1.1.4 Interaction CCXML <-> VoiceXML

Les applications en langage de balisage extensible de commande d'appel (CCXML, *Call Control eXtensible Markup Language*) 1.0 [CCXML10] fournissent des services principalement à travers le contrôle de l'interaction entre les connexions, les conférences, et les dialogues. Bien que CCXML soit capable de prendre en charge des environnements de dialogue arbitraires, VoiceXML est couramment utilisé comme environnement de dialogue en conjonction avec les applications CCXML ; CCXML est spécifiquement conçu pour prendre en charge effectivement l'utilisation de VoiceXML. CCXML 1.0 définit des éléments de langage qui permettent que les dialogues soient préparés, lancés, et terminés ; il permet de plus que les données à retourner par l'environnement de dialogue pour les transferts d'appel soient demandés (par le dialogue) et qu'il y soit répondu par l'application CCXML, et pour des événements arbitraires entre l'application CCXML et l'application de dialogue courante.

L'interface décrite dans le présent document peut être utilisée par les mises en œuvre de CCXML 1.0 pour contrôler les serveurs de supports VoiceXML. Noter cependant que certaines caractéristiques du langage CCXML exigent des facilités de traiter les événements entre CCXML et les sessions VoiceXML qui vont au delà de ce qui est défini dans la présente spécification. Par exemple, les transferts d'appel contrôlés par VoiceXML et les événements de mi-dialogue, définis par l'application ne peuvent pas être pleinement réalisés en utilisant cette seule spécification. Un paquetage d'événements SIP [RFC3265] PEUT être utilisé en plus de la présente spécification pour fournir un traitement d'événements étendu.

1.1.5 Autres cas d'utilisation

En plus des cas d'utilisation décrits ci-dessus, il y a un certain nombre d'autres cas d'utilisation prévus qui ne sont pas décrits en détails, comme :

1. L'utilisation d'un serveur de supports VoiceXML comme adjonction de commutateur privé/distributeur d'appels automatique (PBX/ACD, *Private Branch Exchange / Automatic Call Distributor*) fondé sur IP, éventuellement pour fournir de la messagerie vocale, un participant automatisé, ou d'autres capacités.
2. L'invocation et le contrôle d'une session VoiceXML qui fournit le composant de modalité vocale dans un système multi modal.

1.2 Terminologie

Serveur d'application : un serveur d'application SIP héberge et exécute des services, en particulier en terminant les sessions SIP sur un serveur de supports. Le serveur d'application PEUT aussi agir comme serveur HTTP [RFC2616] en interaction avec les serveurs de supports.

Serveur de supports VoiceXML : un interpréteur VoiceXML incluant un contexte d'interpréteur fondé sur SIP et les capacités de traitement des supports requises pour prendre en charge la fonction VoiceXML.

Session VoiceXML : c'est une session multimédia comportant au moins un agent d'utilisateur SIP, un serveur de supports VoiceXML, les flux de données entre eux, et une application VoiceXML exécutante.

Dialogue VoiceXML : équivalent d'une session VoiceXML.

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

2. Établissement et terminaison de session VoiceXML

Cette Section décrit comment établir une session VoiceXML, avec ou sans préparation, et comment terminer une session. Elle traite aussi comment les informations de session sont rendues disponibles aux applications VoiceXML.

2.1 Identification de service

La demande d'URI SIP est utilisée pour identifier le service de supports VoiceXML. La partie utilisateur de l'URI de demande SIP est fixée à "dialog". Cela est fait pour assurer la compatibilité avec la [RFC4240], car le présent document étend l'interface de dialogue définie dans cette spécification et parce que cette convention de la [RFC4240] est largement adoptée par les serveurs de supports existants.

La normalisation de l'URI de demande SIP incluant la partie utilisateur améliore aussi l'interopérabilité entre serveurs d'application et serveurs de supports, et réduit les frais généraux de provisionnement qui seraient nécessaires si l'utilisation d'un serveur de supports par un serveur d'application exigeait un URI provisionné individuellement. À cet égard, le présent document (et la [RFC4240]) n'ajoute pas de signification à la partie utilisateur, mais normalise plutôt la façon dont sont provisionnées les cibles sur les serveurs de supports. De plus, comme les serveurs d'application -- et non des êtres humains -- sont généralement les clients des serveurs de supports, des problèmes comme l'interprétation et l'internationalisation ne s'appliquent pas.

Exposer un service de supports VoiceXML avec une adresse bien connue peut améliorer la possibilité d'exploitation : il est RECOMMANDÉ que le serveur de supports VoiceXML utilise les mécanismes standard de SIP pour authentifier les points d'extrémité comme discuté à la Section 9.

Le document initial VoiceXML est spécifié avec le paramètre "voicexml". De plus, des paramètres sont définis qui contrôlent comment le serveur de supports VoiceXML va chercher le document VoiceXML spécifié. La liste des paramètres définis par la présente spécification est comme suit (noter que les noms de paramètre sont insensibles à la casse):

voicexml : URI du document initial VoiceXML à aller chercher. Il va normalement contenir un URI HTTP, mais peut utiliser d'autres schémas d'URI, par exemple, pour se référer à des documents VoiceXML locaux, statiques. Si le paramètre "voicexml" est omis, le serveur de supports VoiceXML peut choisir le document VoiceXML initial par d'autres moyens, comme d'appliquer une valeur par défaut, ou il peut rejeter la demande.

maxage : utilisé pour régler la valeur max-age de l'en-tête Cache-Control en conjonction avec les documents VoiceXML cherchés en utilisant HTTP, conformément à la [RFC2616]. Si il est omis, le serveur de supports VoiceXML va utiliser une valeur par défaut.

maxstale : utilisé pour régler la valeur max-stale de l'en-tête Cache-Control en conjonction avec les documents VoiceXML cherchés en utilisant HTTP, conformément à la [RFC2616]. Si il est omis, le serveur de supports VoiceXML va utiliser une valeur par défaut.

method: utilisé pour régler la méthode HTTP appliquée pour aller chercher le document VoiceXML initial. Les valeurs permises sont "get" ou "post" (insensible à la casse). Par défaut "get".

postbody : utilisé pour régler le corps HTTP codé "application/x-www-form-urlencoded" [HTML4] pour les demandes "post" (ou est autrement ignoré).

ccxml : utilisé pour spécifier une "valeur JSON" [RFC4627] qui est transposée en une variable "session.connection.ccxml" VoiceXML -- voir le paragraphe 2.4.

aai : utilisé pour spécifier une "valeur JSON" [RFC4627] qui est transposée en la variable de session "session.connection.aai" VoiceXML -- voir le paragraphe 2.4.

D'autres paramètres spécifiques d'application peuvent être ajoutés à l'URI de demande et sont exposés dans les variables de session VoiceXML (voir le paragraphe 2.4).

Formellement, l'URI de demande pour le service de supports VoiceXML a une partie utilisateur fixe de "dialog". Sept paramètres d'URI sont définis (voir la définition de uri-parameter au paragraphe 25.1 de la [RFC3261]).

```
dialog-param = "voicexml=" vxml-url      ; vxml-url suit l'URI      ; syntaxe définie dans la [RFC3986]
maxage-param  = "maxage=" 1*DIGIT
maxstale-param = "maxstale=" 1*DIGIT
method-param  = "method=" ("get" / "post")
postbody-param = "postbody=" token
ccxml-param   = "ccxml=" json-value
```

aai-param = "aai=" json-value
 json-value = false / null / true / object / array / number / string définis dans la [RFC4627]

Les paramètres de l'URI de demande dans les re-INVITE suivants sont ignorés. Une conséquence de cela est que le serveur de supports VoiceXML ne peut pas recevoir l'instruction par le serveur d'application de changer l'application VoiceXML exécutante après qu'une session VoiceXML a démarré.

Les caractères spéciaux contenus dans les valeurs de dialog-param, postbody-param, ccxml-param, et aai-param doivent être codés en URL ("échappés") comme exigé par la syntaxe d'URI SIP, par exemple, '?' (%3f), '=' (%3d), et ';' (%3b). Le serveur de supports VoiceXML DOIT donc "déséchapper" ces valeurs de paramètre avant de les utiliser ou de les exposer aux applications VoiceXML courantes. Il est important que le serveur de supports VoiceXML déséchappe seulement les valeurs de paramètre une fois que la valeur d'URI VoiceXML désirée pourrait elle-même être codée en URL, par exemple.

Comme certaines applications peuvent choisir de transférer des informations confidentielles, le serveur de supports VoiceXML DOIT prendre en charge le schéma sips: comme discuté à la Section 9.

Note d'information : à l'égard de la valeur postbody-param, comme le contenu application/x-www-form-urlencoded échappe lui-même les caractères non alphanumériques en insérant des %HH de remplacement, les règles d'échappement ci-dessus vont résulter en ce que les caractères '%' soient encore échappés en plus des séparateurs de nom/valeur "&" et "=".

Par exemple, l'URI de demande SIP suivant identifie l'utilisation des services de supports VoiceXML, avec 'http://appserver.exemple.com/promptcollect.vxml' comme document VoiceXML initial, à aller chercher avec les valeurs max-age/max-stale de 3600s/0s, respectivement :

```
sip:dialog@mediaserver.exemple.com; \voicexml=http://appserver.exemple.com/promptcollect.vxml; \
maxage=3600;maxstale=0
```

2.2 Initiation d'une session VoiceXML

Une session VoiceXML est initiée via l'utilisation par le serveur d'application d'une INVITE SIP. Normalement, le serveur d'application va être spécialisé dans la fourniture de services VoiceXML. Au minimum, le serveur d'application peut se comporter comme un simple mandataire en réécrivant l'URI de demande reçu de l'agent d'utilisateur en un URI de demande convenable pour la consommation du serveur de supports VoiceXML (comme spécifié au paragraphe 2.1). Par exemple, un agent d'utilisateur pourrait présenter un numéro : tél :+1-201-555-0123 que le serveur d'application transpose en une application d'assistance à l'annuaire sur le serveur de supports VoiceXML avec un URI de demande de :

```
sip:dialog@ms1.exemple.com; \voicexml=http://as1.exemple.com/da.vxml
```

Certaines valeurs d'en-tête dans le message INVITE au serveur de supports VoiceXML sont transposées en variables de session VoiceXML et sont spécifiées au paragraphe 2.4.

À réception de l'INVITE, le serveur de supports VoiceXML produit une réponse provisoire, 100 Trying, et commence à aller chercher le document VoiceXML initial. La réponse 200 OK indique que le document VoiceXML a été trouvé et analysé correctement et est prêt pour exécution. L'exécution d'application commence à réception du ACK (sauf si le dialogue est préparé comme spécifié au paragraphe 2.3). Noter que la réponse 100 Trying va généralement être envoyée à réception de l'INVITE en accord avec la [RFC3261], car le serveur de supports VoiceXML ne peut pas en général garantir que la collecte initiale va s'achever en moins de 200 ms. Cependant, certaines mises en œuvre peuvent être capables de garantir des temps de réponse à l'INVITE initial, et donc peuvent n'avoir pas besoin d'envoyer une réponse 100 Trying.

Comme optimisation, avant d'envoyer la réponse 200 OK, le serveur de supports VoiceXML PEUT exécuter l'application jusque au point du premier état d'attente ou de purge de l'invite VoiceXML.

Un serveur de supports VoiceXML, comme tout agent d'utilisateur SIP, peut être incapable d'accepter la demande INVITE pour diverses raisons. Par exemple, une offre du protocole de description de session (SDP, *Session Description Protocol*) contenue dans l'INVITE pourrait exiger l'utilisation de codecs qui ne sont pas pris en charge par le serveur de supports. Dans ce cas, le serveur de supports devrait répondre comme défini dans la [RFC3261]. Cependant, il y a des conditions d'erreur spécifiques de VoiceXML, comme suit :

1. Si l'URI de demande ne se conforme pas à la présente spécification, un 400 Mauvaise demande DOIT être retourné

(sauf si il est utilisé pour choisir d'autres services non définis par la présente spécification).

2. Si un paramètre d'URI dans l'URI de demande est répété, alors la demande DOIT être rejeté avec une réponse 400 Mauvaise demande.
3. Si l'URI de demande n'inclut pas de paramètre "voicexml", et si le serveur de supports VoiceXML ne choisit pas d'utiliser une page par défaut, le serveur de supports VoiceXML DOIT retourner une réponse finale de 400 Mauvaise demande, et il DEVRAIT inclure un en-tête Warning avec un code à trois chiffres de 399 et un message d'erreur lisible par l'homme.
4. Si on ne peut pas aller chercher ou analyser le document VoiceXML, le serveur de supports VoiceXML DOIT retourner une réponse finale de 500 Erreur interne du serveur et DEVRAIT inclure un en-tête Warning avec un code à trois chiffres de 399 et un message d'erreur lisible par l'homme.

Note d'information : certaines applications peuvent passer une quantité significative de données au dialogue VoiceXML sous la forme de paramètres d'URI de demande. Cela peut faire que la taille totale de la demande INVITE excède la MTU du réseau sous-jacent. Dans ce cas, les applications/mises en œuvre doivent veiller à utiliser un transport approprié à ces plus grands messages (comme TCP) ou à utiliser d'autres moyens de passer les informations requises au dialogue VoiceXML (comme de fournir un identifiant univoque de session dans l'URI VoiceXML initial et d'utiliser plus tard cet identifiant comme clé pour restituer les données du serveur HTTP).

2.3 Préparation d'une session VoiceXML

Dans certains scénarios, il est bénéfique de préparer une session VoiceXML avant de la faire fonctionner. Une session VoiceXML préparée antérieurement est supposée s'exécuter avec un délai minimal quand on lui dit de le faire.

Si un dialogue SIP sans supports est établi avec le INVITE initial au serveur de supports VoiceXML, l'application VoiceXML ne va pas s'exécuter après la réception de l'ACK. Pour faire fonctionner l'application VoiceXML, le serveur d'application doit produire un re-INVITE pour établir une session de supports.

Un dialogue SIP sans supports peut être établi en envoyant un SDP ne contenant pas de lignes de supports dans l'INVITE initial. Autrement, si aucun SDP n'est envoyé dans le INVITE initial, le serveur de supports VoiceXML va inclure une offre dans le message 200 OK, à laquelle on peut répondre avec une réponse dans le ACK avec le ou les accès de supports réglés à 0.

Une fois que fonctionne une application VoiceXML, un re-INVITE qui désactive le flux de supports (c'est-à-dire, règle les accès à 0) ne va pas autrement affecter l'application exécutante (excepté que les actions de reconnaissance initiées pendant que les flux de supports sont désactivés vont résulter en temporisations sans entrées).

2.4 Transpositions de variable de session

Les variables standard de session VoiceXML ont leurs valeurs allouées conformément à :

`session.connection.local.uri` : s'évalue en l'URI SIP spécifié dans l'en-tête To: de l'INVITE initial.

`session.connection.remote.uri` : s'évalue en l'URI SIP spécifié dans l'en-tête From: de l'INVITE initial.

`session.connection.redirect` : ce dispositif est rempli par des informations contenues dans l'en-tête History-Info [RFC4244] dans l'INVITE initial ou est autrement indéfini. Chaque entrée (hi-entry) dans l'en-tête History-Info est transposée, en ordre inverse, en un élément du dispositif `session.connection.redirect`. Les propriétés de chaque élément du dispositif sont déterminées comme suit :

- * `uri` : réglé à la valeur `hi-targeted-to-uri` de l'entrée History-Info ;
- * `pi` : réglé à "vrai" si `hi-targeted-to-uri` contient un paramètre "Privacy=history", ou si l'en-tête d'INVITE "Privacy" inclut "history"; "faux" autrement ;
- * `if` : réglé à la valeur du paramètre "if" si il existe, indéfini autrement ;
- * `reason` : réglé mot pour mot à la valeur du paramètre "Reason" de `hi-targeted-to-uri`

`session.connection.protocol.name` : s'évalue à "sip". Noter que ceci est destiné à refléter l'utilisation de SIP en général, et ne distingue pas si le serveur de supports a été accédé via les procédures SIP ou SIPS.

`session.connection.protocol.version` : s'évalue à "2.0".

`session.connection.protocol.sip.headers` : c'est un dispositif associatif où chaque clé dans le dispositif est le nom non compact d'un en-tête SIP dans l'INVITE initial converti en minuscules (noter que la conversion de casse ne s'applique pas à la valeur de l'en-tête). Si plusieurs champs d'en-tête de même nom de champ sont présents, les valeurs sont combinées en une seule valeur séparées de virgules. Les mises en œuvre DOIVENT au minimum inclure l'en-tête Call-ID et PEUVENT inclure d'autres en-têtes. Par exemple, `session.connection.protocol.sip.headers["call-id"]` s'évalue à l'identifiant d'appel du dialogue SIP.

`session.connection.protocol.sip.requesturi` : c'est un dispositif associatif où les clés et valeurs du dispositif sont formées à partir des paramètres d'URI sur l'URI de demande SIP de l'INVITE initial. La clé de dispositif est le nom de paramètre d'URI converti en minuscules (noter que la conversion de casse ne s'applique pas à la valeur du paramètre). La valeur de dispositif correspondante est obtenue en évaluant la valeur du paramètre d'URI comme une "valeur JSON" [RFC4627] dans le cas des valeurs `ccxml-param` et `aai-param` et autrement comme une chaîne. De plus, la fonction `toString()` du dispositif retourne l'URI de demande SIP complet. Par exemple, si on suppose un URI de demande de `sip:dialog@exemple.com;voicexml=http://exemple.com;aai=%7b"x":1%2c"y":true%7d` alors

<code>session.connection.protocol.sip.requesturi["voicexml"]</code>	s'évalue	à	"http://exemple.com",
<code>session.connection.protocol.sip.requesturi["aai"].x</code>	s'évalue	à	1 (numéro de type)
<code>session.connection.protocol.sip.requesturi["aai"].y</code>	s'évalue	à	vrai (type Booléen) et
<code>session.connection.protocol.sip.requesturi</code>	s'évalue	à	l'URI de demande complet (type String)

'sip:dialog@exemple.com;voicexml=http://exemple.com;aai={"x":1,"y":true}'.

`session.connection.aai` : s'évalue à `session.connection.protocol.sip.requesturi["aai"]`.

`session.connection.ccxml` : s'évalue à `session.connection.protocol.sip.requesturi["ccxml"]`.

`session.connection.protocol.sip.media` : ceci est un dispositif où chaque élément du dispositif est un objet avec les propriétés suivantes :

- * `type` : cette propriété exigée indique le type du support associé au flux. La valeur est une chaîne. Il est fortement recommandé que les valeurs suivantes soient utilisées pour les types de supports courants : "audio" pour le support audio, et "video" pour le support vidéo.
- * `direction` : cette propriété exigée indique la direction du support par rapport au `session.connection.originator`. Les valeurs définies sont "sendrecv", "sendonly", "recvonly", et "inactive".
- * `format` : cette propriété est facultative. Si elle est définie, la valeur de la propriété est un dispositif. Chaque élément du dispositif est un objet qui spécifie des informations sur un format du support (il y a un élément de support pour chaque type de charge utile sur la ligne m-). L'objet contient au moins une propriété appelée "name" dont la valeur est le sous type MIME du format de support (les sous types MIME sont enregistrés dans la [RFC4855]). D'autres propriétés peuvent être définies avec des valeurs de chaîne ; elles correspondent aux paramètres exigés et, si il en est de définis, facultatifs, du format.

Comme conséquence de cette définition, il y a une entrée de dispositif dans `session.connection.protocol.sip.media` pour chaque ligne m- non désactivée pour la session de supports négociée. Noter que cette variable de session est mise à jour si les caractéristiques de la session de supports changent pour la session VoiceXML (c'est-à-dire, du fait d'un re-INVITE). Par exemple, considérons une connexion avec une loi mu G.711 bidirectionnelle "audio" échantillonnée à 8 kHz. Dans ce cas, `session.connection.protocol.sip.media[0].type` s'évalue à "audio", `session.connection.protocol.sip.media[0].direction` à "sendrecv", `session.connection.protocol.sip.media[0].format[0].name` s'évalue à "audio/PCMU", et `session.connection.protocol.sip.media[0].format[0].rate` s'évalue à "8000".

Noter que quand elles accèdent aux en-têtes SIP et aux paramètres d'URI de demande via les dispositifs associatifs `session.connection.protocol.sip.headers` et `session.connection.protocol.sip.requesturi` définis ci-dessus, les applications peuvent choisir entre deux façons sémantiquement équivalentes de se référer au dispositif. Par exemple, l'une ou l'autre des formes suivantes peut être utilisée pour accéder au paramètre d'URI de demande nommé "foo" :

```
session.connection.protocol.sip.requesturi["foo"]
session.connection.protocol.sip.requesturi.foo
```

Cependant, il est important de noter que tous les noms d'en-tête SIP ou noms de paramètre d'URI de demande ne sont pas des identifiants ECMAScript valides, et à ce titre, peuvent seulement être accédés en utilisant la première forme (notation matricielle). Par exemple, l'en-tête Call-ID peut seulement être accédé comme `session.connection.protocol.sip.headers["call-id"]` ; tenter d'accéder à la même valeur par `session.connection.protocol.sip.headers.call-id` résulterait en une erreur.

2.5 Terminaison d'une session VoiceXML

Le serveur d'application peut terminer une session VoiceXML en produisant un BYE au serveur de supports VoiceXML. À réception d'un BYE dans le contexte d'une session VoiceXML existante, le serveur de supports VoiceXML DOIT envoyer une réponse 200 OK et DOIT lancer un événement "connection.disconnect.hangup" à l'application VoiceXML. Si l'en-tête Reason [RFC3326] est présent dans la demande BYE, alors la valeur de l'en-tête Reason est fournie telle qu'elle via la variable "_message" au sein de la portée de la variable anonyme de l'élément qui la capture.

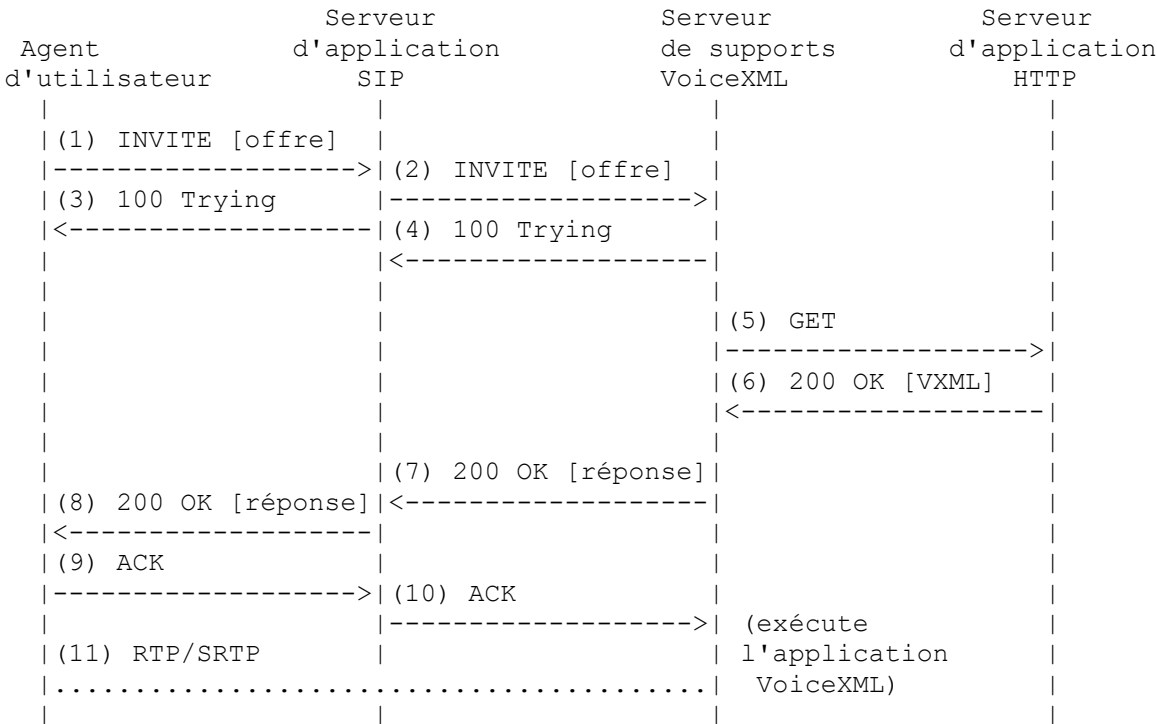
Le serveur de supports VoiceXML peut aussi initier la terminaison de la session en produisant une demande BYE. Cela va normalement se produire par suite de l'apparition d'un <disconnect> ou <exit> dans l'application VoiceXML, du fait que l'application VoiceXML est arrivée à sa fin, ou d'erreurs non traitées au sein de l'application VoiceXML.

Voir à la Section 4 les mécanismes pour retourner les données au serveur d'application.

2.6 Exemples

2.6.1 Établissement de session de base

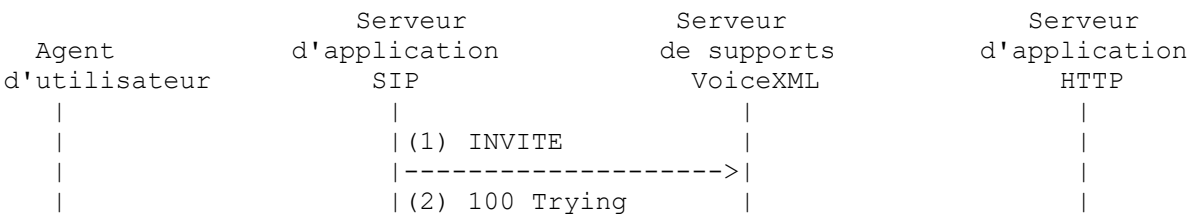
Cet exemple illustre un serveur d'application établissant une session VoiceXML au nom d'un agent d'utilisateur.

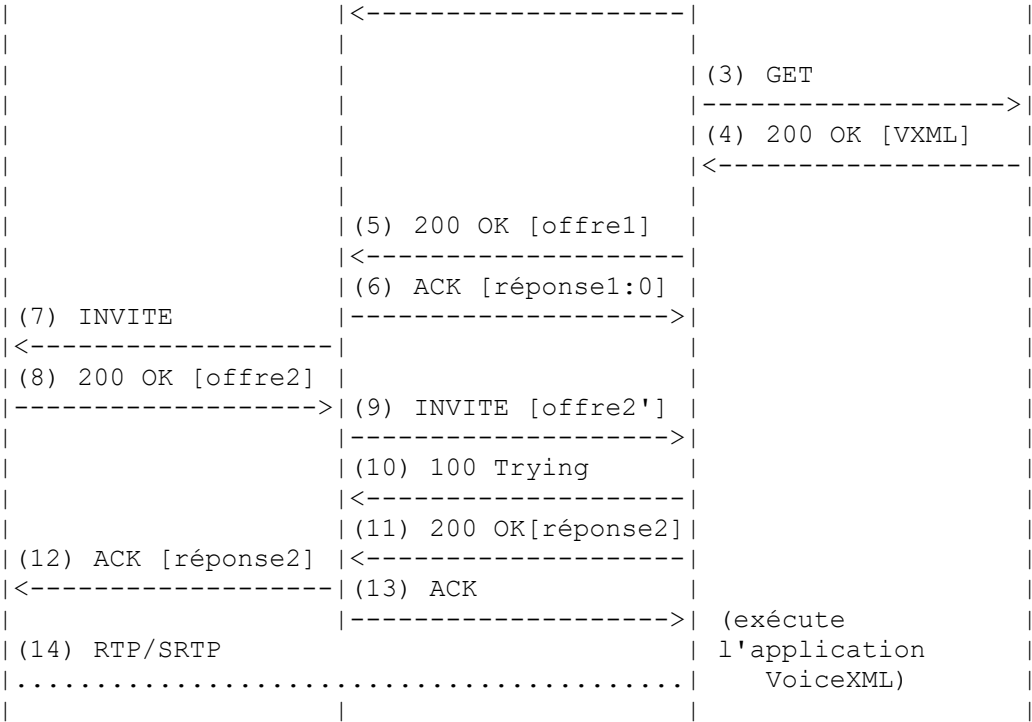


2.6.2 Préparation de session VoiceXML

Cet exemple montre la préparation d'une session VoiceXML. Dans cet exemple, la session VoiceXML est préparée avant de passer un appel sortant à un agent d'utilisateur, et est démarrée aussitôt que l'agent d'utilisateur répond.

La notation [réponse1:0] est utilisée pour indiquer une réponse SDP avec les accès de supports réglés à 0.



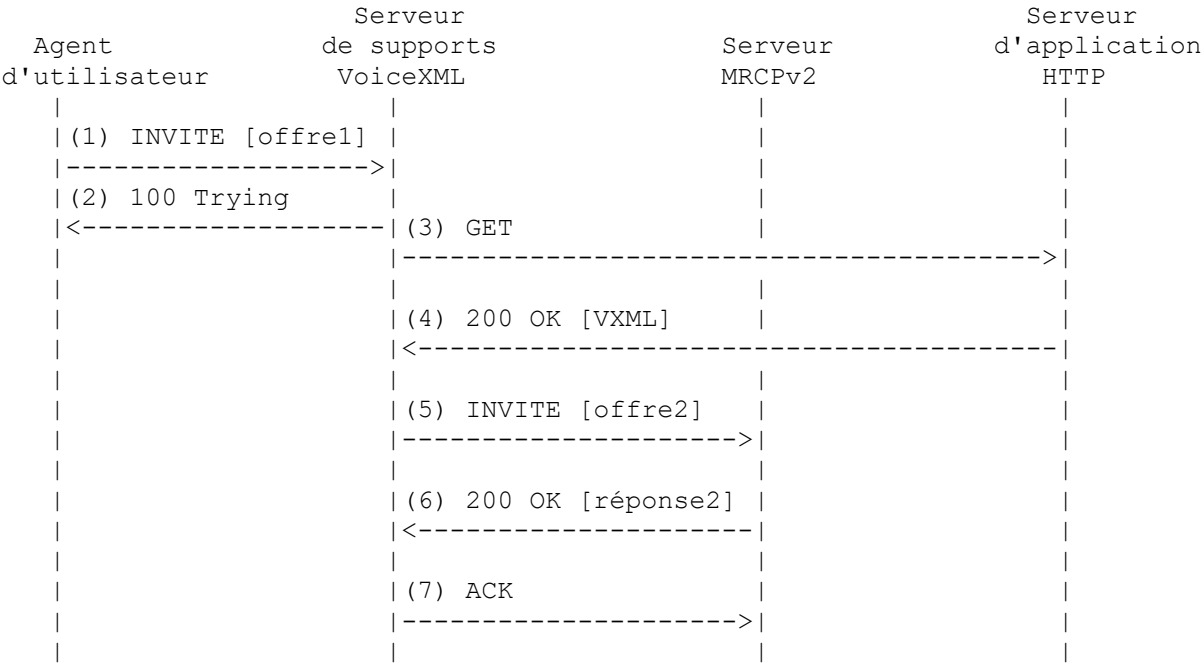


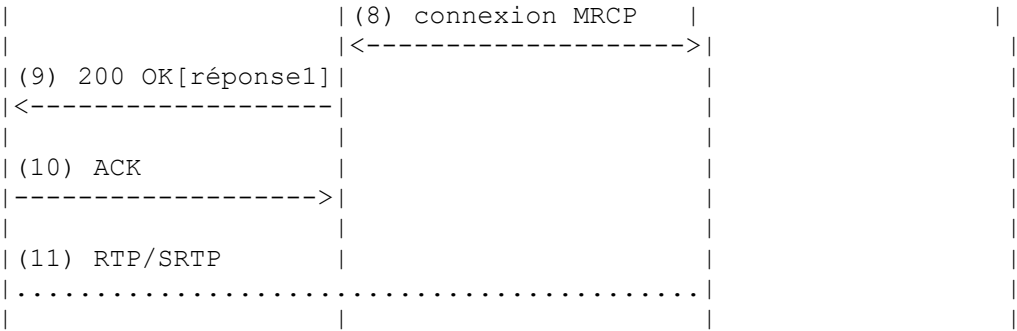
Détail de mise en œuvre : offre2' est déduit de offre2 -- elle duplique les lignes m- et les lignes a- de l'offre2. Cependant, offre2' diffère de offre2 car elle doit contenir la même ligne o- qu'utilisée dans réponse1:0 mais avec le numéro de version incrémenté. Aussi, si offre1 a plus de lignes m- que offre2, alors offre2' doit être bourrée avec des lignes m-supplémentaires (rejetées).

2.6.3 Établissement de MRCP

Le protocole de contrôle de ressources de support (MRCP, *Media Resource Control Protocol*) [RFC6787] est un protocole qui permet à des clients comme un serveur de supports VoiceXML de contrôler les ressources de service de support comme des synthétiseurs de parole, des reconnaissseurs, vérificateurs, et identifieurs résidants dans les serveurs sur le réseau.

L'exemple ci-dessous illustre comment un serveur de supports VoiceXML peut établir une session MRCP en réponse à une INVITE initiale.





Dans cet exemple, le serveur de supports VoiceXML est responsable de l'établissement d'une session avec le serveur de ressource de support MRCPv2 avant d'envoyer la réponse 200 OK à l'INVITE initiale. Le serveur de supports VoiceXML va effectuer l'offre/réponse appropriée avec le serveur de ressource de support MRCPv2 sur la base des capacités SDP du serveur d'application et du serveur de ressource de support MRCPv2. Le serveur de supports VoiceXML va changer l'offre reçue de l'étape 1 pour établir une session MRCPv2 dans l'étape (5) et va réécrire le SDP pour inclure une ligne m- pour chaque ressource MRCPv2 à utiliser et les autres modifications SDP requises comme spécifié par MRCPv2. Une fois que le serveur de supports VoiceXML a effectué l'offre/réponse avec le serveur de ressource de support MRCPv2, il va établir un canal de contrôle MRCPv2 dans l'étape (8). La ressource MRCPv2 est désallouée quand le serveur de supports VoiceXML reçoit ou envoie un BYE (non montré).

3. Prise en charge des supports

Cette Section décrit la prise en charge obligatoire et facultative de supports requise de cette interface.

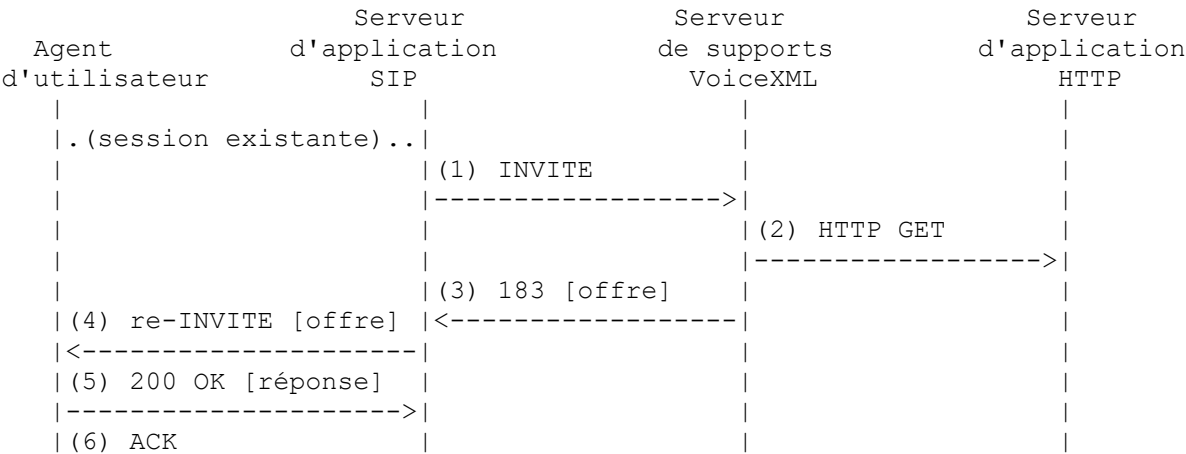
3.1 Offre/réponse

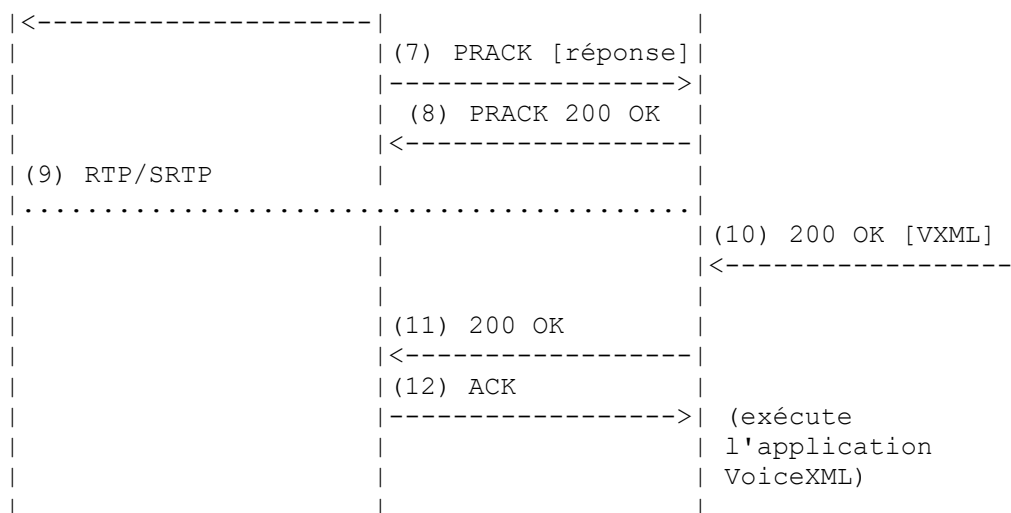
Le serveur de supports VoiceXML DOIT prendre en charge le mécanisme standard d'offre/réponse de la [RFC3264]. En particulier, si une offre SDP n'est pas présente dans l'INVITE, le serveur de supports VoiceXML va faire une offre dans la réponse 200 OK faisant la liste des codecs pris en charge.

3.2 Support précoce

Le serveur de supports VoiceXML PEUT prendre en charge l'établissement précoce de flux de supports comme décrit dans la [RFC3960]. Cela permet au serveur d'application d'établir des flux de supports entre un agent d'utilisateur et le serveur de supports VoiceXML en parallèle avec le document VoiceXML initial traité (qui peut impliquer une génération dynamique de page VoiceXML et une interaction avec des bases de données ou d'autres systèmes). Ceci est utile principalement pour minimiser le délai de démarrage d'une session VoiceXML, particulièrement dans les cas où une session avec l'agent d'utilisateur existe déjà mais où le flux de supports associé à cette session doit être redirigé sur un serveur de supports VoiceXML.

Le flux suivant montre l'utilisation de support précoce (en utilisant le modèle de passerelle défini dans la [RFC3960]) :





Bien que la [RFC3960] préfère utiliser le modèle de serveur d'application pour les supports précoces plutôt que le modèle de passerelle, le principal problème avec le modèle de passerelle -- le fourchement -- est significativement moins courant quant il produit des demandes aux serveurs de supports VoiceXML. C'est parce que les serveurs de supports VoiceXML répondent à toutes les demandes avec des réponses 200 OK en l'absence d'erreurs inhabituelles, et ils font normalement ainsi en plusieurs centaines de millisecondes. Cela fait d'eux des cibles improbables de scénarios de fourchement, car d'autres cibles du processus de fourchement ne vont virtuellement jamais être capables de répondre plus rapidement qu'un système automatique, sauf si ils sont eux-mêmes des systèmes automatiques -- auquel cas, il n'y pas de raison d'établir une compétition de réponse entre deux systèmes automatiques. Les problèmes de génération des tonalités d'appel dans le modèle de passerelle sont aussi atténués, à la fois par le temps de réponse normalement rapide du 200 OK, et parce que la présente spécification oblige qu'aucun paquet de support ne soit généré avant la réception d'un ACK (éliminant donc le besoin que l'agent d'utilisateur effectue l'analyse du paquet de supports).

Noter que l'offre de support précoce par un serveur de supports VoiceXML n'implique pas que l'application VoiceXML référencée puisse toujours être allée cherchée et exécutée avec succès. Par exemple, si le serveur d'application HTTP devait retourner une réponse 4xx à l'étape 10 ci-dessus, ou si le contenu VoiceXML fourni n'était pas valide, le serveur de supports VoiceXML retournerait quand même une réponse 500 (conformément au paragraphe 2.2). À ce point, il serait de la responsabilité du serveur d'application de supprimer tout flux de supports établi avec le serveur de supports.

3.3 Modification de la session de supports

Le serveur de supports VoiceXML DOIT permettre la modification de la session de supports via une re-INVITE et DEVRAIT prendre en charge la méthode UPDATE [RFC3311] pour le même objet. En particulier, il DOIT être possible de changer les flux entre sendrecv, sendonly, et recvonly comme spécifié dans la [RFC3264].

Les flux unidirectionnels sont utiles pour les annonces ou l'écoute seule (mot clé). Le mécanisme préféré pour mettre la session de supports en garde est spécifié dans la [RFC3264], c'est-à-dire, l'UA modifie le flux pour être en envoi seul et change son propre flux. Une modification de la session de supports n'affecte pas l'exécution de l'application VoiceXML (sauf que les actions de reconnaissance initiées pendant le maintien en garde vont résulter en temporisations sans entrée).

3.4 Codecs audio et vidéo

Afin de réaliser un niveau de base d'interopérabilité, ce paragraphe spécifie un sous ensemble minimal de codecs et de formats de charge utile RTP [RFC3550] qui DOIVENT être pris en charge par le serveur de supports VoiceXML.

Pour les applications audio seul, les lois mu et A de G.711 DOIVENT être prises en charge en utilisant les types de charge utile RTP 0 et 8 [RFC3551]. D'autres codecs et formats de charge utile PEUVENT être pris en charge.

Les applications de vidéo téléphonie, qui emploient un flux de vidéo en plus du flux audio, sont possibles dans VoiceXML 2.0/2.1 par l'utilisation de formats de conteneur de fichier multimédia comme les formats .3gp [TS26244] et .mp4 [IEC14496-14]. La prise en charge de la vidéo est facultative pour la présente spécification. Si la vidéo est prise en charge alors :

1. H.263 de base [RFC4629] DOIT être pris en charge. Pour des raisons historiques, la version 1996 de H.263 PEUT être prise en charge en utilisant le format de charge utile RTP défini dans la [RFC2190] (type de charge utile 34 [RFC3551]).
2. L'audio adaptatif multi-débits (AMR) bande étroite [RFC4867] DEVRAIT être pris en charge.
3. La vidéo MPEG-4 [RFC3016] DEVRAIT être prise en charge.
4. L'audio MPEG-4 en codage audio évolué (AAC, *Advanced Audio Coding*) [RFC3016] DEVRAIT être pris en charge.
5. D'autres codecs et formats de charge utile PEUVENT être pris en charge.

Les opérations d'enregistrement vidéo effectuées par le serveur de supports VoiceXML exigent normalement la réception d'une intra-trame avant que l'enregistrement puisse commencer. Le serveur de supports VoiceXML DEVRAIT utiliser le mécanisme décrit dans la [RFC4585] pour demander qu'une nouvelle intra-trame soit envoyée.

Comme certaines applications peuvent choisir de transférer des informations confidentielles, le serveur de supports VoiceXML DOIT prendre en charge Secure RTP (SRTP) [RFC3711] comme discuté à la Section 9.

3.5 DTMF

Les événements DTMF [RFC4733] DOIVENT être pris en charge. Quand l'agent d'utilisateur n'indique pas la prise en charge de la [RFC4733], le serveur de supports VoiceXML PEUT effectuer la détection DTMF en utilisant d'autres moyens comme de détecter les tonalités DTMF dans le flux audio. Note de mise en œuvre : la raison pour laquelle seulement les événements de téléphonie [RFC4733] doivent être utilisés quand l'agent d'utilisateur en indique la prise en charge est d'éviter le risque de double détection du DTMF si la détection sur le flux audio est appliquée simultanément.

4. Retour de données au serveur d'application

Cette section discute les mécanismes pour retourner les données (par exemple, expression collectée ou informations numériques) provenant du serveur de supports VoiceXML au serveur d'application.

4.1 Mécanisme HTTP

À tout moment durant l'exécution de l'application VoiceXML, des données peuvent être retournées au serveur d'application via HTTP en utilisant des éléments VoiceXML standard comme <submit> ou <subdialog>. Notamment, l'élément <data> dans VoiceXML 2.1 [VXML21] permet que les données soient envoyées efficacement au serveur d'application sans exiger une transition de page VoiceXML et est idéal pour les courtes applications VoiceXML comme "prompt and collect".

Pour la plupart des applications, il est nécessaire de corréler les informations passées sur HTTP avec une session VoiceXML particulière. Une façon de réaliser cela est d'inclure le Call-ID SIP (accessible dans VoiceXML via le dispositif `session.connection.protocol.sip.headers`) dans les champs POST HTTP. Autrement, un unique "URI POST-back" peut être spécifié comme un paramètre d'URI spécifique d'application dans l'URI de demande de l'INVITE initial (accessible dans VoiceXML via le dispositif `session.connection.protocol.sip.requesturi`).

Comme certaines applications peuvent choisir de transférer des informations confidentielles, le serveur de supports VoiceXML DOIT prendre en charge le schéma https: comme discuté à la Section 9.

4.2 Mécanisme SIP

Des données peuvent être retournées au serveur d'application via l'attribut `expr` ou `namelist` sur <exit> ou l'attribut `namelist` sur <disconnect>. Un serveur de supports VoiceXML DOIT prendre en charge le codage des données `expr/namelist` dans le corps de message d'une demande BYE envoyée du serveur de supports VoiceXML par suite de l'élément <exit> ou <disconnect>. Un serveur de supports VoiceXML PEUT prendre en charge l'inclusion de données `expr/namelist` dans le corps de message du message 200 OK en réponse à une demande BYE reçue (c'est-à-dire, quand l'application VoiceXML répond à l'événement `session.disconnect.hangup` et ensuite exécute un élément <exit> avec l'attribut `expr` ou `namelist` spécifié).

Noter que l'envoi de données `expr/namelist` dans la réponse 200 OK exige que le serveur de supports VoiceXML retarde la réponse finale à la demande BYE reçue jusqu'à ce que se termine l'état de traitement final post déconnexion de l'application VoiceXML. Ce mécanisme est soumis à la contrainte que le serveur de supports VoiceXML réponde avant que le temporisateur F du client d'agent d'utilisateur (UAC) arrive à expiration (par défaut, 32 secondes). De plus, pour les transports non fiables, l'UAC va retransmettre la demande BYE conformément aux règles de la [RFC3261]. Le serveur de supports VoiceXML DEVRAIT mettre en œuvre les recommandations de la [RFC4320] sur quand envoyer la réponse provisoire 100 Trying à la demande BYE.

Si une application VoiceXML exécute un `<disconnect>` [VXML21] et ensuite exécute un `<exit>` avec des informations de `namelist`, les informations de `namelist` provenant de l'élément `<exit>` sont éliminées.

Les variables `namelist` sont d'abord converties en leur "valeur JSON" équivalente [RFC4627] et codées dans le corps de message en utilisant le type de contenu de format `application/x-www-form-urlencoded` [HTML4]. Le comportement résultant de la spécification d'une variable d'enregistrement dans la `namelist` ou d'un objet ECMAScript avec des références circulaires n'est pas défini. Si l'attribut `"expr"` est spécifié sur l'élément `<exit>` au lieu de l'attribut `namelist`, le nom réservé `__exit` est utilisé.

Pour permettre au serveur d'application de différencier entre un BYE résultant d'un `<disconnect>` d'un résultant d'un `<exit>`, le nom réservé `__reason` est utilisé, avec une valeur de `"disconnect"` (sans les guillemets) pour refléter l'utilisation de l'élément `<disconnect>` de VoiceXML, et une valeur de `"exit"` (sans les guillemets) à un `<exit>` explicite dans le document VoiceXML. Si la session se termine pour d'autres raisons (comme lorsque le serveur de supports rencontre une erreur) ce paramètre peut être omis, ou peut prendre des valeurs spécifiques de la plate-forme préfixées par un souligné.

La présente spécification étend le `application/x-www-form-urlencoded` en remplaçant les caractères non ASCII par un ou plusieurs octets de la représentation UTF-8 du caractère, avec chaque octet remplacé à son tour par `%HH`, où HH représente la notation hexadécimale en majuscules de la valeur d'octet et `%` est un caractère littéral. Par conséquent, le champ d'en-tête `Content-Type` dans un message BYE contenant des données `expr/namelist` DOIT être réglé à `application/x-www-form-urlencoded; charset=utf-8`.

Le tableau suivant donne des exemples d'usage de `<exit>` et du contenu du résultat correspondant.

Usage de <code><exit></code>	Contenu résultant
<code><exit/></code>	<code>__reason=exit</code>
<code><exit expr="5"/></code>	<code>__exit=5&__reason=exit</code>
<code><exit expr="done"/></code>	<code>__exit="done"&__reason=exit</code>
<code><exit expr="userAuthorized"/></code>	<code>__exit=true&__reason=exit</code>
<code><exit namelist="pin errOrs"/></code>	<code>pin=1234&errOrs=0&__reason=exit</code>

en supposant les variables et valeurs VoiceXML suivantes :

```
userAuthorized = vrai
pin = 1234
errors = 0
```

Par exemple, considérons le sous programme VoiceXML :

```
...
<exit namelist="id pin"/>
...
```

Si `id` égal 1234 et `pin` égal 9999, le message BYE ressemblerait à :

```
BYE sip:user@pc33.exemple.com SIP/2.0
Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
Max-Forwards: 70
From: sip:dialog@exemple.com;tag=a6c85cf
To: sip:user@exemple.com;tag=1928301774
Call-ID: a84b4c76c66710
CSeq: 231 BYE
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: 30
```

id=1234&pin=9999&__reason=exit

Comme certaines applications peuvent choisir de transférer des informations confidentielles, le serveur de supports VoiceXML DOIT prendre en charge le codage S/MIME du corps de message SIP comme discuté à la Section 9.

5. Appel sortant

Les appels sortants peuvent être déclenchés via le serveur d'application en utilisant le contrôle d'appel de tiers [RFC3725].

Le flux IV de la [RFC3725] est recommandé en conjonction avec le mécanisme de préparation de session VoiceXML. Ce flux a plusieurs avantages sur les autres, à savoir :

1. Le choix d'un serveur de supports VoiceXML et la préparation de l'application VoiceXML peuvent se produire avant que l'appel soit placé pour éviter que l'appelé subisse des délais.
2. Éviter des difficultés de coordination qui pourraient survenir avec d'autres flux à cause du temps pris pour aller chercher et analyser le document VoiceXML initial.
3. Le flux est compatible IPv6.

Un exemple de flux pour un appel sortant initié par le serveur d'application est donné au paragraphe 2.6.2.

6. Transfert d'appel

Bien que le cœur de VoiceXML soit un langage de dialogue, il fournit aussi une capacité facultative de transfert d'appel. La capacité de transfert de VoiceXML convient particulièrement au cas d'utilisation de nœud de service IVR du RTPC décrit au paragraphe 1.1.2. Il n'est PAS RECOMMANDÉ d'utiliser la capacité de transfert de VoiceXML dans les réseaux qui impliquent des serveurs d'application. Le serveur d'application lui-même peut plutôt fournir la fonction d'acheminement d'appel en faisant des actions de signalisation sur la base des données qui lui sont retournées du serveur de supports VoiceXML via HTTP ou dans le message BYE de SIP.

Si le transfert VoiceXML est pris en charge, le mécanisme décrit dans cette section DOIT être employé. Les flux de transfert spécifiés ici sont choisis sur la base de la fourniture du meilleur interfonctionnement sur une large gamme d'appareils SIP. Les mises en œuvre de CCXML<->VoiceXML, qui exigent un couplage étroit sous la forme d'événements bidirectionnels pour prendre en charge tous les types de transferts définis dans VoiceXML, peuvent bénéficier d'autres approches, comme l'utilisation de paquetages d'événements SIP [RFC3265].

Dans ce qui suit, les réponses provisoires ont été omise au nom de la clarté de la présentation.

6.1 Blind

La séquence blind-transfer (*transfert aveugle*) est initiée par le serveur de supports VoiceXML via un message REFER [RFC3515] dans le dialogue SIP original. L'en-tête Refer-To contient l'URI pour l'appelé, comme spécifié via les attributs dest ou destexpr sur l'étiquette VoiceXML <transfer>.

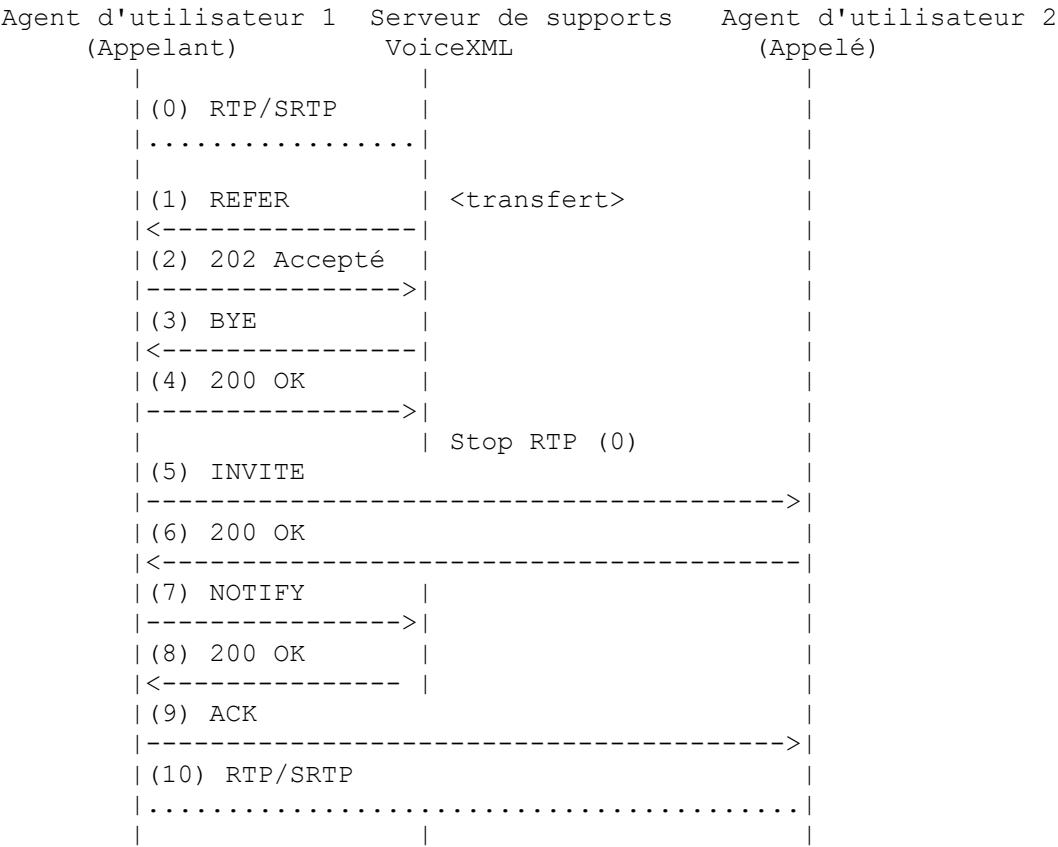
Si la demande REFER est acceptée, et dans ce cas le serveur de supports VoiceXML va recevoir une réponse 2xx, le serveur de supports VoiceXML lance l'événement connection.disconnect.transfer et va terminer la session VoiceXML avec un message BYE. Pour les transferts aveugles, les mises en œuvre PEUVENT utiliser la [RFC4488] pour supprimer l'abonnement implicite associé au message REFER.

Si la demande REFER résulte en une réponse non 2xx, la variable d'élément de forme du <transfer> (ou l'événement soulevé) dépend de la réponse SIP et est spécifiée dans le tableau suivant. Noter que cela indique que la demande de transfert a été rejetée.

Réponse SIP	variable/événement <transfer>
404 Pas trouvé	error.connection.baddestination
405 Méthode interdite	error.unsupported.transfer.blind

503 Service indisponible error.connection.noresource
(Pas de réponse) network_busy
(Autre 3xx/4xx/5xx/6xx) inconnu

Un exemple est illustré ci-dessous (les réponses provisoires et les messages NOTIFY correspondant aux réponses provisoires ont été omis pour plus de clarté).



Si l'attribut aai ou aaiexpr est présent sur <transfert>, il est ajouté à l'URI Refer-To comme paramètre nommé "aai" dans la méthode REFER. Les caractères réservés sont codés en URL comme requis pour les URI SIP/SIPS [RFC3261]. La transposition des valeurs en dehors de la gamme ASCII est spécifique de la plate-forme.

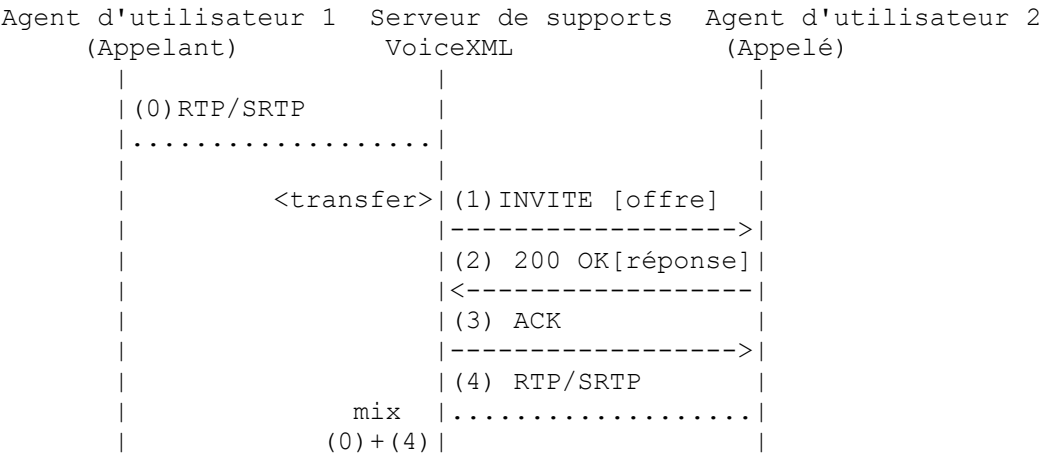
6.2 Bridge

La fonction de transfert "bridge" (*pont*) résulte en la création d'une petite session multi-parties impliquant l'appelant, le serveur de supports VoiceXML, et l'appelé. Le serveur de supports VoiceXML invite l'appelé à la session et va éjecter l'appelé si le transfert est terminé.

Si l'attribut aai ou aaiexpr est présent sur <transfer>, il est ajouté à l'URI de demande dans le INVITE comme paramètre d'URI nommé "aai". Les caractères réservés sont codés en URL comme requis pour les URI SIP/SIPS [RFC3261]. La transposition des valeurs en dehors de la gamme ASCII est spécifique de la plate-forme.

Durant la tentative de transfert, l'audio spécifié dans l'attribut transferaudio de <transfert> est envoyé à l'agent d'utilisateur 1. Un serveur de supports VoiceXML PEUT exécuter les supports précoces reçus de l'appelé à l'appelant si l'attribut transferaudio est omis.

La séquence de transfert "bridge" est illustrée ci-dessous. Le serveur de supports VoiceXML (agissant comme un UAC) fait un appel à l'agent d'utilisateur 2 avec les mêmes codecs qu'utilisés par l'agent d'utilisateur 1. Quand l'établissement d'appel est terminé, RTP s'écoule entre l'agent d'utilisateur 2 et le serveur de supports VoiceXML. Ce flux est mélangé avec celui de l'agent d'utilisateur 1'.



Si une réponse finale à l'INVITE n'est pas reçue de l'agent d'utilisateur 2 et si le temporisateur de connexion arrive à expiration (spécifié comme dans un attribut de <transfert>) le serveur de supports VoiceXML va produire un CANCEL pour terminer la transaction et la variable d'élément de forme de <transfert> est réglée à noanswer (*pas de réponse*).

Si l'INVITE résulte en une réponse non 2xx, la variable d'élément de forme de <transfert> (ou l'événement soulevé) dépend de la réponse SIP et est spécifié dans le tableau suivant.

Réponse SIP	variable/événement <transfer>
404 Non trouvé	error.connection.baddestination
405 Méthode interdite	error.unsupported.transfer.bridge
408 Fin de temporisation de la demande	pas de réponse
486 Occupé ici	occupé
503 Service indisponible	error.connection.noresource
(Pas de réponse)	network_busy
(Autre 3xx/4xx/5xx/6xx)	inconnu

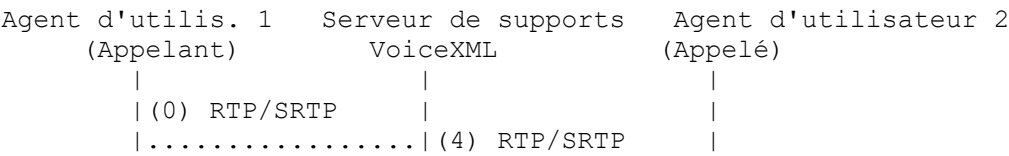
Une fois le transfert établi, le serveur de supports VoiceXML peut "écouter" sur le flux de supports venant de l'agent d'utilisateur 1 pour effectuer de la parole ou un mot-clé DTMF, qui quand il correspond résulte en la déconnexion de l'extrémité proche, c'est-à-dire que le serveur de supports VoiceXML produit un BYE à l'agent d'utilisateur 2 et que l'application VoiceXML continue avec l'agent d'utilisateur 1. Un BYE va aussi être produit à l'agent d'utilisateur 2 si la durée de l'appel dépasse le maximum spécifié dans l'attribut maxtime sur <transfert>.

Si l'agent d'utilisateur 2 produit un BYE durant le transfert, le transfert se termine et la variable d'élément de forme du <transfert> VoiceXML reçoit la valeur de "far_end_disconnect". Si l'agent d'utilisateur 1 produit un BYE durant le transfert, le transfert se termine et l'événement VoiceXML "connection.disconnect.transfer" est lancé.

6.3 Consultation

Le transfert de consultation (aussi appelé transfert attendu [RFC5359]) est similaire à un transfert aveugle sauf que le résultat de l'établissement de l'appel de transfert est connu et que l'appelant n'est pas éliminé par suite de l'échec du transfert.

Le transfert de consultation commence avec le même flux que pour le transfert de pont sauf que les flux RTP ne sont pas mélangés à l'étape (4) et que error.unsupported.transfer.consultation supplante error.unsupported.transfer.bridge. En supposant qu'un nouveau dialogue SIP avec l'agent d'utilisateur 2 est créé, le reste de la séquence suit comme illustré ci-dessous (les réponses provisoires et les messages NOTIFY correspondant aux réponses provisoires ont été omis pour plus de clarté). Le transfert de consultation utilise l'en-tête Replaces: [RFC3891] de sorte que l'agent d'utilisateur 1 appelle l'agent d'utilisateur 2 et remplace le dialogue SIP du premier avec le serveur de supports VoiceXML par un nouveau dialogue SIP entre l'appelant et l'appelé.



```

| ..... |
| (5) REFER |
| <-----> |
| (6) 202 Accepté |
| -----> |
| (7) INVITE Replaces:ms1.exemple.com |
| -----> |
| (8) 200 OK |
| <-----> |
| (9) ACK |
| -----> |
| (10) RTP/SRTP |
| ..... |
| (11) BYE |
| <-----> |
| (12) 200 OK |
| -----> Arrête RTP (4) |
| (13) NOTIFY |
| -----> |
| (14) 200 OK |
| <-----> |
| (15) BYE |
| <-----> |
| (16) 200 OK |
| -----> Arrête RTP (0) |
| |

```

Si une réponse autre que 202 Accepté est reçue en réponse à la demande REFER envoyée à l'agent d'utilisateur 1, le transfert se termine et un événement `error.unsupported.transfer.consultation` est soulevé. De plus, un BYE est envoyé à l'agent d'utilisateur 2 pour terminer la branche sortante établie.

Le serveur de supports VoiceXML utilise la réception d'un message NOTIFY avec un message sipfrag de 200 OK pour déterminer que le transfert de consultation a réussi. Quand cela arrive, l'événement `connection.disconnect.transfer` va être lancé à l'application VoiceXML, et un BYE est envoyé à l'agent d'utilisateur 1 pour terminer la session. Un message NOTIFY avec une réponse finale non 2xx du corps de message sipfrag va résulter en la terminaison du transfert et la variable d'élément d'entrée VoiceXML associée va être réglée à "unknown". Noter qu'en conséquence de ce mécanisme, les mises en œuvre NE DOIVENT PAS utiliser la [RFC4488] pour supprimer l'abonnement implicite associé au message REFER pour les transferts de consultation.

7. Contributeurs

Le gros du travail préparatoire de ce travail a été effectué par des téléconférences hebdomadaires et sur la messagerie. Les auteurs tiennent particulièrement à reconnaître les contributions de R. J. Auburn (Voxeo), Jeff Haynie (Hakano), et Scott McGlashan (Hewlett-Packard).

8. Remerciements

Le présent document doit sa genèse à "Interface SIP aux serveurs de dialogue VoiceXML", dont les auteurs sont J. Rosenberg, P. Mataga, et D. Ladd. Les personnes suivantes ont fait des apports au document actuel : R. J. Auburn (Voxeo), Hans Bjurstrom (Hewlett-Packard), Emily Candell (Comverse), Peter Danielsen (Lucent), Brian Frasca (Tellme), Jeff Haynie (Hakano), Scott McGlashan (Hewlett-Packard), Matt Oshry (Tellme), Rao Surapaneni (Tellme)

Les auteurs tiennent à remercier de leur soutien Cullen Jennings et les présidents de Mediactrl, Eric Burger et Spencer Dawkins.

9. Considérations sur la sécurité

Exposer un service de supports VoiceXML avec une adresse bien connue peut améliorer la possibilité d'exploitation (par exemple, un service réseau invoqué peut déclencher un événement de facturation). Il est RECOMMANDÉ que le serveur de supports VoiceXML utilise les mécanismes SIP standard [RFC3261] pour authentifier les points d'extrémité demandeurs et les autorise selon la politique locale.

Certaines applications peuvent choisir de transférer des informations confidentielles au ou du serveur de supports VoiceXML. Pour assurer la confidentialité des données, le serveur de supports VoiceXML DOIT mettre en œuvre les schémas sips: et https: en plus du codage de corps de message S/MIME comme décrit dans la [RFC3261].

Le serveur de supports VoiceXML DOIT prendre en charge Secure RTP (SRTP) [RFC3711] pour assurer la confidentialité, l'authentification, et la protection contre la répétition pour les flux de supports RTP (incluant le trafic de contrôle RTCP).

Pour atténuer la possibilité d'attaques de déni de service, il est RECOMMANDÉ que le serveur de supports VoiceXML (en plus de l'authentification et l'autorisation des points d'extrémité décrits ci-dessus) fournisse des mécanismes pour mettre en œuvre les politiques locales comme la limitation du temps d'exécution de l'application VoiceXML.

10. Considérations relatives à l'IANA

L'IANA a enregistré les paramètres suivants dans le registre des paramètres d'URI SIP/SIPS, suivant la politique de spécification exigée de la [RFC3969] :

Nom de paramètre	Valeurs prédéfinies	Référence
maxage	Non	RFC 5552
maxstale	Non	RFC 5552
method	"get" / "post"	RFC 5552
postbody	Non	RFC 5552
ccxml	Non	RFC 5552
aai	Non	RFC 5552

11. Références

11.1 Références normatives

- [HTML4] Raggett, D., Le Hors, A., and I. Jacobs, "HTML 4.01 Specification", W3C Recommendation, déc. 1999.
- [RFC2119] S. Bradner, "[Mots clés à utiliser](#) dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. DOI 10.17487/RFC2119, (*MàJ par RFC8174*)
- [RFC2616] R. Fielding et autres, "[Protocole de transfert hypertexte](#) -- HTTP/1.1", juin 1999. (*D.S., MàJ par 2817, 6585*)
- [RFC3016] Y. Kikuchi et autres, "Format de charge utile RTP pour flux audio/vidéo MPEG-4", novembre 2000. (*Obs., voir RFC6416*) (*P.S.*)
- [RFC3261] J. Rosenberg et autres, "SIP : [Protocole d'initialisation de session](#)", juin 2002, DOI 10.17487/RFC3261. (*Mise à jour par 3265, 3853, 4320, 4916, 5393, 6665, 8217, 8760*)
- [RFC3264] J. Rosenberg et H. Schulzrinne, "[Modèle d'offre/réponse](#) avec le protocole de description de session (SDP)", juin 2002, DOI 10.17487/RFC3264. (*P.S. ; MàJ par RFC8843, 9143*)
- [RFC3265] A.B. Roach, "[Notification d'événement spécifique](#) du protocole d'initialisation de session (SIP)", juin 2002. (*MàJ par RFC6446*) (*Remplacée par la RFC6665*)
- [RFC3311] J. Rosenberg, "[Méthode UPDATE](#) du protocole d'initialisation de session (SIP) ", octobre 2002.
- [RFC3326] H. Schulzrinne, D. Oran, G. Camarillo, "[Champ d'en-tête Reason](#) pour le protocole d'initialisation de session

(SIP)", décembre 2002. (P.S. : MàJ par RFC9366)

- [RFC3515] R. Sparks, "[Méthode Refer](#) du protocole d'initialisation de session (SIP)", avril 2003. (MàJ par RFC8217)
- [RFC3550] H. Schulzrinne, S. Casner, R. Frederick et V. Jacobson, "[RTP : un protocole de transport pour les applications en temps réel](#)", STD 64, DOI 10.17487/RFC3550, juillet 2003. (MàJ par RFC7164, RFC7160, RFC8083, RFC8108, RFC8860)
- [RFC3551] H. Schulzrinne et S. Casner, "[Profil RTP pour conférences audio](#) et vidéo avec contrôle minimal", STD 65, juillet 2003. (MàJ par RFC8860)
- [RFC3711] M. Baugher et autres, "Protocole de [transport sécurisé en temps réel](#) (SRTP)", mars 2004, DOI 10.17487/RFC3711. (P.S. ; MàJ par RFC9335)
- [RFC3725] J. Rosenberg et autres, "Bonnes pratiques actuelles [pour la commande d'appel de tiers \(3pcc\)](#) dans le protocole d'initialisation de session (SIP)", avril 2004, DOI 10.17487/RFC3725. (BCP0085)
- [RFC3891] R. Mahy, B. Biggs, R. Dean, "[En-tête 'Replaces'](#) du protocole d'initialisation de session (SIP)", septembre 2004. (P.S.)
- [RFC3986] T. Berners-Lee, R. Fielding et L. Masinter, "[Identifiant de ressource uniforme](#) (URI) : Syntaxe générique", STD 66, janvier 2005. (P.S. ; MàJ par RFC8820)
- [RFC4244] M. Barnes, éd., "Extension au protocole d'initialisation de session (SIP) pour les informations d'historique de demande", novembre 2005. (P.S.) (Remplacée par RFC7044)
- [RFC4320] R. Sparks, "Actions visant les [problèmes identifiés dans la transaction Non-INVITE](#) du protocole d'initialisation de session (SIP)", janvier 2006. (MàJ RFC3261) (P.S.)
- [RFC4488] O. Levin, "[Suppression de l'abonnement implicite](#) de la méthode REFER du protocole d'initialisation de session (SIP)", mai 2006. (P.S.)
- [RFC4585] J. Ott et autres, "[Profil RTP étendu pour rétroaction](#) fondée sur le protocole de contrôle de transport en temps réel (RTCP) (RTP/AVPF)", juillet 2006. (P.S., MàJ par RFC8108)
- [RFC4627] D. Crockford, "Type de support application/json pour la notation d'objet JavaScript (JSON)", juillet 2006. (Information ; remplacée par RFC7159)
- [RFC4629] J. Ott et autres, "Format de charge utile RTP pour la vidéo de la Recommandation UIT-T H.263", janvier 2007. (Remplace RFC2429) (MàJ RFC3555) (P.S.)
- [RFC4733] H. Schulzrinne, T. Taylor, "[Charge utile RTP pour chiffres DTMF](#), tonalités téléphoniques, et signaux de téléphonie", décembre 2006. (Remplace RFC2833) (MàJ par RFC4734, RFC5244) (P.S.)
- [RFC4855] S. Casner, "[Enregistrement du type de support](#) des formats de charge utile RTP", février 2007. (P.S.)
- [RFC4867] J. Sjöberg et autres, "[Format de charge utile RTP](#) et format de mémorisation de fichier pour les codecs audio multi débit adaptif (AMR) et multi débit adaptif large bande (AMR-WB)", avril 2007. (Remplace RFC3267) (P.S.)
- [VXML20] McGlashan, S., Burnett, D., Carter, J., Danielsen, P., Ferrans, J., Hunt, A., Lucas, B., Porter, B., Rehor, K., and S. Tryphonas, "Voice Extensible Markup Language (VoiceXML) Version 2.0", W3C Recommendation, mars 2004.
- [VXML21] Oshry, M., Auburn, R J., Baggia, P., Bodell, M., Burke, D., Burnett, D., Candell, E., Kilic, H., McGlashan, S., Lee, A., Porter, B., and K. Rehor, "Voice Extensible Markup Language (VoiceXML) Version 2.1", W3C Candidate Recommendation, juin 2005.

11.2 Références pour information

- [CCXML10] Auburn, R J., "Voice Browser Call Control: CCXML Version 1.0", W3C Working Draft, juin 2005.
- [IEC14496-14] ISO/CEI 14496-14:2003, "Technologie de l'information - Codage des objets audio-visuels – Format de fichier MP4", octobre 2003.
- [RFC2190] C. Zhu, "Format de charge utile RTP pour les flux vidéo H.263", septembre 1997. (*Historique, voir RFC4628*)
- [RFC3960] G. Camarillo, H. Schulzrinne, "[Support précoce et génération des tonalités d'appel](#) dans le protocole d'initialisation de session (SIP)", décembre 2004. (*Information*)
- [RFC3969] G. Camarillo, "[Registre des paramètres d'identifiant](#) de ressource uniforme (URI) de l'IANA pour le protocole d'initialisation de session (SIP)", décembre 2004. ([BCP0099](#) ; *Mis à jour par RFC 5727*)
- [RFC4240] E. Burger et autres, "Services de base de support réseau avec SIP", décembre 2005. (*Information*)
- [RFC5359] A. Johnston et autres, "Exemples de services avec le protocole d'initialisation de session", octobre 2008. ([BCP0144](#))
- [RFC6787] D. Burnett, S. Shanmugham, "Protocole de contrôle des ressource du support version 2 (MRCPv2)", nov. 2012. (*P.S.*)
- [TS23002] 3rd Generation Partnership Project, "Network architecture (Release 6)", 3GPP TS 23.002 v6.6.0, décembre 2004.
- [TS26244] 3GPP TS 26.244 v6.4.0, "Transparent end-to-end paquet switched streaming service (PSS); 3GPP file format (3GP)", décembre 2004.

Appendice A. Notes sur les références normatives

On a fait une référence normative "dégradée" à la [RFC4627] -- un document pour information qui décrit un format propriétaire (mais extrêmement populaire).

Adresse des auteurs

Dave Burke
Google
Belgrave House, 76 Buckingham Palace Road
London SW1W 9TQ
United Kingdom
mél : daveburke@google.com

Mark Scott
Genesys
1120 Finch Avenue West, 8th floor
Toronto, Ontario M3J 3H7
Canada
mél : Mark.Scott@genesyslab.com