Groupe de travail Réseau
Request for Comments : 4730
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

E. Burger, Cantata Technology, Inc.
M. Dolly, AT&T Labs
novembre 2006

Paquetage d'événement du protocole d'initialisation de session (SIP) pour stimulus clavier (KPML)

Statut du présent mémoire
Le présent document spécifie un protocole de l'Internet en cours de 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.

Résumé
Le présent document décrit un paquetage d'événements SIP "kpml" qui permet de surveiller les signaux de multi fréquences bi tonalités (DTMF, *Dual Tone Multi-Frequency*) et utilise des documents de langage de balisage extensible (XML, *Extensible Markup Language*) auxquels on se réfère sous le nom de langage de balisage pour stimulus clavier (KPLM, *Key Press Markup Language*). Le paquetage d'événements kpml peut être utilisé pour prendre en charge des applications cohérentes avec les principes définis dans le document intitulé "Cadre pour l'interaction d'application dans le protocole d'initialisation de session (SIP)". Le paquetage d'événement utilise les messages SUBSCRIBE et permet les documents XML qui définissent et décrivent des spécifications de filtre pour capturer des frappes de touches de clavier (tonalités DTMF) entrées à l'interface d'utilisateur de présentation libre de l'agent d'utilisateur (UA, *User Agent*) SIP. Le paquetage d'événements utilise les messages NOTIFY et permet aux documents XML de rapporter les touches de clavier capturées (tonalités DTMF) en cohérence avec les spécifications de filtre, à un serveur d'application. La portée de ce paquetage est de collecter des frappes de touches supplémentaires ou des frappes de touches à mi appel (déclencheurs).

Table des matières

1. Introduction

Le présent document décrit un paquetage d'événements SIP "kpml" qui permet de surveiller la frappe des touches et utilise les documents XML désignés comme de langage de balisage pour stimulus clavier (KPML, *Key Press Markup Language*). KPML est un balisage [XML2] qui permet des interfaces d'utilisateur de présentation libre comme décrit dans le cadre d'interaction d'application de la [RFC5629]. Le paquetage de stimulus de frappe de touche du clavier est un paquetage de notification d'événement SIP [RFC3265] qui utilise les méthodes SUBSCRIBE et NOTIFY de SIP. Les corps de filtre d'abonnement et de rapport de notification utilisent le langage de balisage de stimulus clavier (KPML, *Keypad Markup Language)*.

Le paquetage d'événements "kpml" exige la définition de deux nouveaux types MIME, deux nouveaux sous espaces de noms d'URN, et deux schémas pour la demande KPML et la réponse KPML. La portée de ce paquetage est de collecter les frappes de touches supplémentaires ou les frappes de touches à mi appel (déclencheurs). Cette capacité permet à un fournisseur de service de serveur d'application de surveiller (filtrer) un ensemble de schémas DTMF chez un agent d'utilisateur SIP situé soit dans un appareil d'utilisateur d'extrémité, soit dans une passerelle.

En particulier, le paquetage d'événements "kpml" permet aux "téléphones sourds" et aux "passerelles" qui reçoivent des signaux provenant de téléphones sourds de faire rapport des événements de frappe de touches de clavier d'utilisateur. Familièrement, ce mécanisme assure le "rapport des chiffres" ou le "rapport de multi fréquence bi-tonalités (DTMF)". Cette capacité élimine le besoin "de faire faire une épingle à cheveux" (acheminer le support dans puis hors du même appareil) à travers un serveur de supports ou de dupliquer tous les événements DTMF, quand un serveur d'application a besoin de déclencher un traitement de mi appel sur des schémas de chiffres DTMF.

Un objectif de KPML est de tenir dans une mémoire et une empreinte de traitement extrêmement petites.

Le nom du document XML, KPML, reflète son rôle de support traditionnel. Le réseau téléphonique public commuté (RTPC) réalise la signalisation en transportant les tonalités DTMF dans le canal support (signalisation dans la bande) du terminal de l'utilisateur au commutateur local.

Les réseaux de voix sur IP transportent les signaux dans la bande avec les fréquences DTMF réelles ou des paquets de la [RFC2833]. Dans la RFC 2833, l'application de signalisation insère les paquets de signalisation désignés comme RFC 2833

aussi bien que, ou au lieu, de générer des tonalités dans le chemin des supports. L'application receveuse reçoit les informations de signalisation dans le flux de supports.

Les tonalités de la RFC 2833 sont idéales pour convoyer des événements téléphoniques en point à point dans un flux de protocole de transport en temps réel (RTP, *Real-time Transport Protocol*) car dans le contexte de sessions directes comme un appel entre deux parties ou une conférence simple, mixée centralement. Cependant, il y a d'autres environnements où des exigences supplémentaires ou de remplacement sont nécessaires. Ces autres environnements incluent des traductions de protocole et un contrôle d'appel complexe.

Une application intéressée pourrait demander la notification de chaque frappe de touche. Cependant, beaucoup des cas d'utilisation pour une telle signalisation montrent que la plupart des applications sont intéressées à seulement une ou quelques frappes de touches. Donc, un mécanisme est nécessaire pour spécifier à l'interface de l'utilisateur quels stimuli l'application exige.

1.1 Conventions du document

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].

Le document de cadre d'interaction d'applications [RFC5629] donne l'interprétation des termes "appareil d'utilisateur", "application SIP", et "entrée d'usager". Le présent document utilise les termes "application" et "application demandeuse" de façon interchangeable avec "application SIP".

De plus, le document de cadre d'interaction d'applications discute des mandataires d'appareil d'utilisateur. Une instanciation courante de mandataire d'appareil d'utilisateur est une passerelle du réseau téléphonique public commuté (RTPC). Parce que le comportement normatif d'une interface d'utilisateur de présentation libre est identique pour un agent d'utilisateur SIP de présentation libre et un mandataire d'appareil d'utilisateur de présentation libre, le présent document utilise "appareil d'utilisateur" dans les deux cas.

2. Vue d'ensemble du protocole

Le paquetage d'événements "kpml" utilise des demandes explicites de notification d'abonnement qui utilisent les méthodes SIP SUBSCRIBE et NOTIFY. Une application qui veut collecter des chiffres crée un document application/kpml-request+xml avec le schéma de chiffres qui l'intéresse et place ce document dans sa demande SUBSCRIBE. Les messages SIP SUBSCRIBE sont acheminés à l'interface d'utilisateur en utilisant l'acheminement standard de demande SIP. Les abonnements KPML ne fourchent pas. La demande KPML contenue dans le message SUBSCRIBE identifie le flux de supports cible en faisant référence aux identifiants de dialogue correspondant à la session responsable du flux de supports. Une fois un abonnement établi, l'interface d'utilisateur envoie des documents application/kpml-response+xml dans des demandes NOTIFY quand les chiffres sont collectés ou quand des fins de temporisation ou des erreurs surviennent.

Un abonnement KPML peut être persistant ou pour une seule fois. Les demandes persistantes sont actives jusqu'à la fin de l'abonnement, jusqu'à ce que l'application remplace la demande, que l'application supprime la demande en envoyant un document nul sur le dialogue, ou que l'application supprime explicitement l'abonnement en envoyant un message SUBSCRIBE avec une valeur d'expiration de zéro (0).

Les demandes pour une seule fois terminent l'abonnement à la réception de valeurs DTMF qui fournissent une correspondance. L'élément KPML "persist" spécifie si l'abonnement reste actif pour la durée spécifiée dans le message SUBSCRIBE ou si il se termine automatiquement sur une correspondance de schéma.

Les messages NOTIFY peuvent contenir des documents XML. Si l'interface d'utilisateur correspond à un schéma de chiffres, le message NOTIFY (de réponse) contient un document XML qui indique l'entrée d'utilisateur détectée et si l'interface d'utilisateur a supprimé la représentation de l'entrée d'utilisateur, comme des tonalités, des données de la RFC 2833, ou des flux de supports. Si l'interface d'utilisateur a rencontré une condition d'erreur, comme une fin de temporisation, cela sera aussi rapporté.

3. Concepts clés

3.1 Durée d'abonnement

KPML reconnaît deux types d'abonnements : pour une fois et persistants. les abonnements persistants ont deux sous types : à notification continue et à notification unique.

Les abonnements pour une seule fois se terminent après que se produit une correspondance de schéma et un rapport est produit dans un message NOTIFY. Si l'interface d'utilisateur détecte un stimulus de pression de touche qui déclenche un événement KPML d'une seule fois, alors l'interface d'utilisateur (notificateur) DOIT régler "l'état d'abonnement" dans le message NOTIFY à "terminé". À ce moment, l'interface d'utilisateur DOIT considérer l'abonnement comme terminé.

Les abonnements persistants restent actifs à l'interface d'utilisateur, même après une correspondance. Pour les abonnements persistants à notification continue, l'interface d'utilisateur va émettre un message NOTIFY chaque fois que l'entrée de l'utilisateur correspond à un schéma d'abonnement. Pour les abonnements persistants à une seule notification, l'appareil utilisateur va émettre un message NOTIFY à la première correspondance, mais n'émettra pas d'autre message NOTIFY jusqu'à ce que l'application produise de nouvelles demandes d'abonnement sur le dialogue d'abonnement.

Note : l'abonnement persistant à une seule notification permet un isolement verrouillé (sans compétition) de l'entrée d'utilisateur entre différentes transpositions de chiffres.

L'attribut "persist" de l'étiquette <pattern> dans le corps de l'abonnement KPML affecte la durée de vie de l'abonnement.

Si l'attribut "persist" est "one-shot", alors une fois qu'il y a eu une correspondance (ou si aucune correspondance n'est possible) l'abonnement se termine après que l'interface d'utilisateur l'a notifié à l'application.

Si l'attribut "persist" est "persist" ou "single-notify", alors l'abonnement se termine quand l'application le termine explicitement ou quand l'interface d'utilisateur termine l'abonnement.

Si l'interface d'utilisateur ne prend pas en charge les abonnements persistants, elle retourne un message NOTIFY avec le code d'état KPML réglé à 531. Si il y a des chiffres dans la mémoire tampon et que les chiffres correspondent à une expression dans le filtre SUBSCRIBE, l'interface d'utilisateur prépare le message de réponse NOTIFY approprié.

Les valeurs de l'attribut "persist" sont sensibles à la casse.

3.2 Temporisateurs

Pour traiter les divers scénarios de collecte de frappe de clés, trois temporisateurs sont définis. Ce sont les temporisateurs extra, critique, et inter chiffres.

- Le temporisateur inter chiffres est le temps maximum d'attente entre les chiffres. Note : à la différence du protocole de contrôle de passerelle de supports (MGCP, *Media Gateway Control Protocol*) [RFC3435] ou de H.248 [RFC3525], il n'y a pas de temporisateur de démarrage, car ce concept ne s'applique pas dans le contexte de KPML.

- Le temporisateur critique est le temps d'attente d'un autre chiffre si les chiffres collectés peuvent correspondre à plus d'un schéma potentiel.

- Le temporisateur extra est le temps d'attente d'un autre chiffre si les chiffres collectés peuvent seulement correspondre à un schéma potentiel, mais une correspondance plus longue est possible pour ce schéma.

L'interface d'utilisateur PEUT prendre en charge une valeur de temporisation entre les chiffres. C'est la durée pendant laquelle l'interface d'utilisateur va attendre une entrée d'utilisateur avant de retourner un résultat d'erreur de fin de temporisation sur un schéma à correspondance partielle. L'application peut spécifier la temporisation inter chiffres comme un nombre entier de millisecondes en utilisant l'attribut "interdigittimer" *(temporisateur inter chiffres)* de l'étiquette <pattern>. Sa valeur par défaut est 4000 millisecondes. Si l'interface d'utilisateur ne prend pas en charge la spécification d'une temporisation inter chiffres, elle DOIT ignorer en silence la spécification. Si l'interface d'utilisateur prend en charge la spécification d'une temporisation inter chiffres, mais pas la granularité spécifiée par la valeur présentée, l'interface d'utilisateur DOIT arrondir la valeur demandée à la plus proche valeur qu'elle peut supporter.

L'objet de la temporisation inter chiffres est d'empêcher les applications de commencer à correspondre à un schéma, et de

ne jamais retourner de résultat. Cela peut se produire, par exemple, si l'utilisateur entre accidentellement une clé qui commence à correspondre à un schéma. Cependant, comme l'utilisateur a entré accidentellement la clé, le reste du schéma ne vient jamais. De plus, quand l'utilisateur entre effectivement un schéma, comme il a déjà entré une clé, le schéma ne peut pas correspondre ou ne peut pas correspondre comme attendu. De même, considérons le cas où l'utilisateur pense avoir entré une frappe de clé, mais l'interface d'utilisateur ne détecte pas la clé. Cela pourrait arriver quand il collecte dix chiffres, mais que l'appareil n'en reçoit en fait que 9. Dans ce cas, l'interface d'utilisateur va attendre éternellement la dixième frappe, alors que l'utilisateur est frustré en se demandant pourquoi l'application ne répond pas.

L'interface d'utilisateur PEUT prendre en charge une valeur de temporisateur de chiffres critiques. C'est la durée pendant laquelle l'interface d'utilisateur va attendre une autre frappe de clé quand il a déjà un <regex> qui correspond mais qu'il y a un autre<regex> plus long qui peut aussi correspondre au schéma. L'application peut spécifier la temporisation de chiffre critique comme un nombre entier de millisecondes en utilisant l'attribut "criticaldigittimer" sur l'étiquette <pattern>. La valeur par défaut est 1000 millisecondes.

L'objet de la temporisation critical-digit est de permettre à l'application de collecter de plus longues correspondances que la plus courte présentée. Ceci est différent de MGCP [RFC3435], où la plus courte correspondance est retournée. Par exemple, si l'application enregistre les schémas "0011", "011", "00", et "0", la temporisation critical-digit permet à l'interface d'utilisateur de distinguer entre "0", "00", "011", et "0011". Sans cette caractéristique, la seule valeur que l'interface d'utilisateur peut détecter est "0".

L'interface d'utilisateur PEUT prendre en charge une valeur de temporisation extra-digit. C'est la durée pendant laquelle l'interface d'utilisateur va attendre une autre frappe de touche quand elle a déjà une correspondance avec le plus long <regex>. L'application peut spécifier la temporisation extra-digit comme un nombre entier de millisecondes en utilisant l'attribut "extradigittimer" sur l'étiquette <pattern>. La valeur par défaut est 500 millisecondes. Si aucune touche d'entrée n'est spécifiée, alors l'interface d'utilisateur PEUT régler par défaut le temporisateur extra-digit à zéro.

L'objet du temporisateur extra-digit est de permettre à l'interface d'utilisateur de collecter les touches d'entrée *(enterkey)*. Sans cette caractéristique, l'interface d'utilisateur aurait la correspondance avec le schéma, et la touche entrée serait mise en mémoire tampon et retournée comme prochain schéma.

3.3 Correspondances de schéma

Pendant la durée de vie de l'abonnement, l'interface d'utilisateur peut détecter un stimulus de frappe de touche qui déclenche un événement KPML. Dans ce cas, l'interface d'utilisateur (notificateur) DOIT retourner le document KPML approprié.

La logique de confrontation de schéma fonctionne comme suit. Les interfaces d'utilisateur KPML DOIVENT suivre la logique présentée dans ce paragraphe afin que des mises en œuvre différentes produisent de façon déterministe le même document KPML étant donnée la même entrée d'utilisateur.

Un document de demande kpml contient un élément <pattern> avec une série d'étiquettes <regex>. Chaque élément <regex> spécifie un schéma potentiel de correspondance pour l'interface d'utilisateur. Le paragraphe 5.1 décrit le langage Dregex *(digit regular expression)* d'expression régulière chiffrée.

L'algorithme de correspondance de schéma recherche la plus longue expression régulière. C'est le même mode que H.248.1 [RFC3525] et non le mode présenté par MGCP [RFC3435]. Le choix de l'algorithme de correspondance de schéma a un impact sur la détermination de quand un schéma correspond. Considérons le document KPML suivant :

```
 <?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
     version="1.0">
  <pattern>
    <regex>0</regex>
    <regex>011</regex>
  </pattern>
</kpml-request>
```

Figure 1 : Correspondance avare

Dans la Figure 1, si on devait avoir la correspondance sur le premier schéma trouvé, la chaîne "011" ne correspondrait jamais parce que la règle "0" va correspondre en premier.

Bien que ce comportement soit ce que la plupart des applications désirent, il a un coût. Considérons le document KPML suivant :

```
<regex>x{7}</regex>
<regex>x{10}</regex>
```

Figure 2 : Correspondance de fin de temporisation

La Figure 2 montre un plan de numérotation Nord Américain typique. Du point de vue d'une application, les utilisateurs attendent un nombre de sept chiffres pour répondre rapidement, sans attendre la temporisation typique inter chiffres critique (généralement de quatre secondes). À l'inverse, l'utilisateur ne veut pas que le système coupe son numéro de dix chiffres à sept chiffres parce qu'il ne l'a pas rentré assez vite.

Une approche de ce problème est d'avoir une terminaison explicite de chaîne de numérotation. Souvent, c'est la touche dièse (#). Maintenant, considérons le fragment suivant :

```
<regex>x{7}#</regex>
<regex>x{10}#</regex>
```

Figure 3 : Temporisation correspondant à l'entrée

Le problème avec l'approche de la Figure 3 est que le "#" va apparaître dans la chaîne de numérotation retournée. De plus, on veut souvent permettre à l'utilisateur d'entrer la chaîne sans la touche de terminaison de la chaîne de numérotation. De plus, utiliser une correspondance explicite sur la touche signifie qu'on doit doubler le nombre de schémas, par exemple, "x{7}", "x{7}#", "x{10}", et "x{10}#".
L'approche utilisée dans KPML est d'avoir une "touche Entrée" explicite, comme montré dans le fragment suivant.

```
<pattern enterkey="#">
  <regex>x{7}</regex>
  <regex>x{10}</regex>
</pattern>
```

Figure 4 : Fin de temporisation correspondant avec la touche Entrée

Dans la Figure 4, l'attribut enterkey à l'étiquette <pattern> spécifie une chaîne qui termine un schéma. Dans cette situation, si l'utilisateur entre sept chiffres suivis par la touche "#", le schéma correspond (ou échoue) immédiatement. KPML indique une non correspondance terminée avec un code d'état KPML de 402.

Note : enterkey est une chaîne. L'attribut enterkey peut être une séquence de touches de clés, comme "**".

Certains schémas cherchent des touches de clés de longue durée. Par exemple, certaines applications attendent un "#" ou "*" long.

KPML utilise le modificateur "L" aux caractères <regex> pour indiquer des longues touches de clés. Le document KPML suivant demande une longue touche de la clé dièse d'au moins 3 secondes.

```
<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
    version="1.0">
 <pattern long="3000">
  <regex>L#</regex>
 </pattern>
</kpml-request>
```

Long dièse

La demande peut spécifier ce qui constitue "long" en réglant l'attribut long dans le <pattern>. Cet attribut est un entier représentant le nombre de millisecondes. Si l'utilisateur presse une clé pendant plus longtemps que "long" millisecondes, le modificateur Long est vrai. La longueur par défaut de l'attribut long est 2500 millisecondes.

Les interfaces d'utilisateur DOIVENT distinguer entre une entrée longue et une entrée courte quand le document KPML spécifie les deux dans un document. Cependant, si il n'y a pas un schéma correspondant à une longue pression de clé dans un document, l'interface d'utilisateur DOIT correspondre au schéma de pression de clé sans considération de la durée de la pression de la clé par l'utilisateur.

À titre d'exemple, dans le fragment suivant de la Figure 6, l'interface d'utilisateur fait la différence entre un "*" long et un "*" normal, mais toute longueur de "#" va correspondre au schéma.

```
<pattern>
  <regex tag="short_star">*</regex>
  <regex tag="long_star">L*</regex>
  <regex>#</regex>
</pattern>
```

Figure 6 : Correspondance longue et courte

Certaines interfaces d'utilisateur sont incapables de présenter de longues pressions de clé. Un exemple est celui d'un appareil téléphonique d'un vieux commutateur privé (PBX, *private branch exchange*) qui émet des tonalités de longueur fixe quand l'utilisateur presse une touche. Pour traiter ce problème, l'interface d'utilisateur PEUT interpréter une succession de pressions d'une seule clé comme équivalente à une longue pression de la même clé. L'application indique qu'elle veut ce comportement en réglant l'attribut "longrepeat" de <pattern> à "vrai".

Le document KPML spécifie si les schémas doivent être persistants en réglant l'attribut "persist" de l'étiquette <pattern> à "persist" ou "single-notify". Toute autre valeur, incluant "one-shot", indique que la demande est un abonnement pour une seule fois. Si l'interface d'utilisateur ne prend pas en charge les abonnements persistants, elle retourne un document KPML avec le code d'état KPML réglé à 531. Si il y a des chiffres dans la mémoire tampon et si les chiffres correspondent à une expression dans le document KPML, l'interface d'utilisateur émet la notification kpml appropriée.

Noter que les valeurs de l'attribut "persist" sont sensibles à la casse.

Certaines interfaces d'utilisateur peuvent prendre en charge plusieurs expressions régulières dans une certaine demande de schéma. Dans cette situation, l'application peut souhaiter savoir quel schéma a déclenché l'événement.

KPML fournit un attribut "tag" à l'étiquette <regex>. "tag" est une chaîne opaque que l'interface d'utilisateur renvoie dans le rapport de notification à l'occasion d'une correspondance dans la transposition de chiffres. Dans le cas de plusieurs correspondances, l'interface d'utilisateur DOIT choisir la plus longue correspondance dans le document KPML. Si plusieurs correspondances ont la même longueur, l'interface d'utilisateur DOIT choisir la première expression mentionnées dans le document d'abonnement KPML sur la base de l'ordre du document KPML.

Si l'interface d'utilisateur ne peut pas prendre en charge plusieurs expressions régulières dans une demande de schéma, l'interface d'utilisateur DOIT retourner un document KPML avec le code d'état KPML réglé à 532. Si l'interface d'utilisateur ne peut pas prendre en charge le nombre d'expressions régulières de la demande de schéma, l'interface d'utilisateur DOIT retourner un document KPML avec le code d'état KPML réglé à 534.

Note : On pourrait rendre obligatoire un nombre minimum d'expressions régulières qu'une interface d'utilisateur doit prendre en charge par demande d'abonnement et globalement. Cependant, un tel minimums tendrait à devenir une limite incorporée. À titre d'indication, on devrait être capable de traiter facilement des dizaines d'expressions par abonnement et des milliers globalement. Une bonne mise en œuvre devrait n'avoir en fait pas de limites. Ceci dit, pour contrer de possibles attaques de déni de service, les mises en œuvre d'interfaces d'utilisateur devraient connaître les codes d'état 534 et 501 et se sentir libres de les utiliser.

3.4 Suppression de chiffres

Dans les conditions de fonctionnement de base, une interface d'utilisateur KPML va transmettre des tonalités dans la bande (de la [RFC2833] ou des tonalités réelles) en parallèle avec le rapport d'entrée d'utilisateur.

Note : Si KPML n'avait pas ce comportement, une interface d'utilisateur exécutant KPML pourrait alors facilement casser les applications appelées. Par exemple, prenons un assistant personnel qui utilise "*9" pour attirer l'attention. Si l'utilisateur presse la touche "*", KPML va garder le chiffre dans l'attente du "9". Que se passe t-il si l'utilisateur entre juste la touche "*", éventuellement parce elle accède à un système de réponse vocale interactive (IVR, *interactive voice response*) qui cherche un "*" ? Dans ce cas, le "*" va être retenu par l'interface d'utilisateur, parce qu'elle cherche le schéma "*9". L'utilisateur va probablement presser à nouveau la touche "*", pensant que le système IVR appelé n'a pas entendu la touche pressée. À ce point, l'interface d'utilisateur va envoyer les deux entrées "*", car "**" ne correspond pas à "*9". Cependant, cela n'aura pas l'effet attendu par l'utilisateur quand il a pressé "*".

Par ailleurs, il y a des situations où passer les tonalités dans la bande n'est pas souhaitable. De telles situations incluent des centres d'appel qui utilisent des départs de tonalités dans la bande pour initier un transfert.

Pour ces situations, KPML ajoute une étiquette de suppression, "pre", à l'étiquette <regex>. Il NE DOIT PAS y avoir plus d'une étiquette <pre> dans une étiquette <regex>.

Si il y a seulement un <pattern> et une seule <regex>, le traitement de suppression est direct. Le point d'extrémité passe l'entrée d'utilisateur jusqu'à ce que le flux corresponde à l'expression <pre> régulière. À ce point, l'interface d'utilisateur va continuer de collecter les entrées d'utilisateur, mais va supprimer la génération ou la transmission de toute entrée d'utilisateur dans la bande.

Si l'interface d'utilisateur a supprimé le stimulus, elle DOIT l'indiquer en incluant l'attribut "suppressed" avec la valeur de "vrai" dans la notification.

Il est clair que si l'interface d'utilisateur traite le document KPML avec l'entrée d'utilisateur mise en mémoire tampon, il est trop tard pour supprimer la transmission de l'entrée d'utilisateur, car l'interface d'utilisateur a depuis longtemps envoyé le stimulus. C'est une situation où il y a une spécification de <pre>, mais l'attribut "suppressed" ne sera pas "vrai" dans la notification. Si il y a une étiquette <pre> qui correspond à l'interface d'utilisateur et si l'interface d'utilisateur est incapable de supprimer l'entrée d'utilisateur, elle DOIT régler l'attribut "suppressed" à "faux".

Une interface d'utilisateur KPML PEUT effectuer une suppression. Si elle n'est pas capable de suppression, elle ignore l'attribut suppression. Elle DOIT régler l'attribut "suppressed" à "faux". Dans ce cas, le schéma qui doit correspondre est le schéma enchaîné de pre+valeur.

À un certain moment, l'interface d'utilisateur va collecter assez d'entrées d'utilisateur pour qu'elles correspondent à un schéma <pre>. L'attribut interdigittimer indique la durée d'attente pour que l'utilisateur entre un stimulus avant de rapporter une erreur de fin de temporisation. Si le temporisateur inter chiffres expire, l'interface d'utilisateur DOIT produire un rapport de fin de temporisation, transmettre l'entrée d'utilisateur supprimée sur le flux de supports, et arrêter la suppression.

Quand l'interface d'utilisateur détecte une correspondance et envoie une demande NOTIFY pour rapporter l'entrée d'utilisateur, elle DOIT arrêter la suppression. En clair, si une entrée d'utilisateur suivante correspond à une autre expression <pre>, alors l'interface d'utilisateur DOIT commencer la suppression.

Après le début de la suppression, il peut devenir clair qu'il n'y aura pas de correspondance. Par exemple, dans l'expression

```
<regex><pre>*8</pre>xxx[2-9]xxxxxx</regex>
```

Au moment où l'interface d'utilisateur reçoit "*8", elle va arrêter de transmettre le stimulus. Disons que les trois chiffres suivants sont "408". Si le chiffre suivant est un zéro ou un un, le schéma ne va pas correspondre.

Note : Il est d'une importance critique pour l'interface d'utilisateur d'avoir un temporisateur inter chiffres sensible. C'est parce qu'un point (".") errant peut supprimer pour toujours l'envoi de chiffres.

Les applications devraient être très attentives à n'indiquer la suppression que quand elles sont bien sûres que l'utilisateur va entrer une chaîne de chiffres qui va correspondre à l'expression régulière. De plus, les applications devraient traiter des situations de non correspondance ou de fin de temporisation. C'est parce que l'interface d'utilisateur va conserver les chiffres, ce qui va évidemment poser des problèmes d'interface d'utilisateur en cas de défaillance.

3.5 Comportement de mémoire tampon d'entrée d'utilisateur

Les interfaces d'utilisateur DOIVENT mettre en mémoire tampon l'entrée d'utilisateur à réception d'un abonnement authentifié et accepté. Les documents KPML suivants appliquent leurs schémas à l'entrée d'utilisateur mise en mémoire tampon. Certaines applications utilisent des interfaces modales où les premières pressions de touches déterminent la signification des pressions de touches suivantes. Pour un utilisateur novice, l'application peut exécuter une invite décrivant dans quel mode est l'application. Cependant, les "utilisateurs expérimentés" ignorent souvent l'invite.

Les interfaces d'utilisateur NE DOIVENT PAS fournir à un abonné des chiffres qui ont été détectés avant l'authentification et l'autorisation de cet abonné. Sans interdiction, un abonné pourrait être capable d'obtenir l'accès à une carte d'appel ou d'autres informations qui empêchent la participation de l'abonné à l'appel. Noter que cette interdiction DOIT être appliquée abonnement par abonnement.

KPML fournit une étiquette <flush> *(purge)* dans l'élément <pattern>. Le comportement par défaut est de ne pas purger l'entrée d'utilisateur. La purge d'entrée d'utilisateur a pour effet d'ignorer les frappes de touches entrées avant l'installation de l'abonnement KPML. Pour purger l'entrée d'utilisateur, il faut inclure l'étiquette <flush>yes</flush> dans le document d'abonnement KPML. Noter que cette directive affecte seulement la combinaison actuelle de dialogue/identifiant de l'abonnement.

Le traitement verrouillé de l'entrée d'utilisateur est que lorsque l'interface d'utilisateur produit une notification, l'application traite la notification alors que l'interface d'utilisateur met en mémoire tampon les entrées d'utilisateur supplémentaires : quand l'application demande plus d'entrées d'utilisateur, et seulement alors, l'interface d'utilisateur le notifie à l'application sur la base des entrées d'utilisateur collectées. Pour conduire l'interface d'utilisateur à opérer en mode verrouillé, régler l'attribut <pattern> à persist="single-notify".

L'interface d'utilisateur DOIT être capable de traiter <flush>no</flush>. Cette directive est effectivement un no-op.

D'autres valeurs de chaîne pour <flush> pourront être définies à l'avenir. Si l'interface d'utilisateur reçoit une chaîne qu'elle ne comprend pas, elle DOIT traiter la chaîne comme un no-op.

Si l'utilisateur presse une touche qui ne peut pas correspondre à un schéma dans l'étiquette <regex>, l'interface d'utilisateur DOIT éliminer toutes les touches de clés en mémoire tampon, jusque et y compris la touche de clé actuelle, de la prise en considération à l'égard des documents KPML actuels ou futurs sur un dialogue donné. Cependant, comme décrit ci-dessus, une fois qu'il y a une correspondance, l'interface d'utilisateur met en mémoire tampon toutes les touches entrées par l'utilisateur à la suite de la correspondance.

Note : Ce comportement permet aux applications de ne recevoir que les entrées d'utilisateur qui les intéressent. Par exemple, une application pré payée souhaite seulement surveiller un long dièse. Si l'utilisateur entre d'autres stimuli, probablement pour d'autres applications, l'application pré payée ne veut pas de notification de cette entrée d'utilisateur. Cette caractéristique est fondamentalement différente du comportement d'un équipement fondé sur le multiplexage à répartition dans le temps où chaque application reçoit chaque touche de clé.

Pour limiter les rapports aux seules correspondances complètes, on établit l'attribut "nopartial" de l'étiquette <pattern> à "vrai". Dans ce cas, l'interface d'utilisateur tente de correspondre à une fenêtre glissante sur l'entrée d'utilisateur collectée.

Les abonnements KPML sont indépendants. Donc, il n'est pas possible au document en cours de savoir si un document suivant va permettre le chargement ou si il veut purger l'entrée d'utilisateur. Donc, l'interface d'utilisateur DOIT mettre en mémoire tampon toutes les entrées d'utilisateur, sous réserve de l'avertissement forced_flush *(purge forcée)* décrit ci-dessous.

Sur un certain dialogue SUBSCRIBE avec un certain identifiant, l'interface d'utilisateur DOIT mettre en mémoire tampon toute entrée d'utilisateur détectée entre le moment du rapport et la réception du prochain document, si il en est. Si le prochain document indique une purge de mémoire tampon, l'interpréteur DOIT purger toutes les entrées d'utilisateur collectées pour être prises en considération à partir des documents KPML reçus sur ce dialogue avec l'identifiant d'événement donné. Si le prochain document n'indique pas la purge des entrées d'utilisateur mises en mémoire tampon, l'interpréteur DOIT alors appliquer l'entrée d'utilisateur collectée (si possible) aux transpositions de chiffres présentées par les étiquettes <regex> du script. Si il y a correspondance, l'interpréteur DOIT suivre les procédures du paragraphe 5.3. Si il n'y en a pas, l'interpréteur DOIT purger toutes les entrées d'utilisateur collectées.

Étant donné le besoin potentiel d'une mémoire tampon infinie pour les entrées d'utilisateur, l'interface d'utilisateur PEUT éliminer les plus anciennes entrées d'utilisateur de la mémoire tampon. Si l'interface d'utilisateur élimine les chiffres, quand

l'interface d'utilisateur produit une notification KPML, elle DOIT régler l'attribut forced_flush *(purge forcée)* de l'étiquette <response> à "vrai". Pour une utilisation future, l'application DOIT considérer toute valeur non nulle, autre que "faux", qu'elle ne comprend pas comme étant la même que "vrai".

Note : L'exigence de mettre en mémoire tampon toutes les entrées d'utilisateur pour toute la durée de la session n'est pas coûteuse en fonctionnement normal. Par exemple, si on a une passerelle avec 8 000 sessions, et si la passerelle met en mémoire tampon 50 frappes de clés sur chaque session, l'exigence est seulement de 400 000 octets, en supposant un octet par touche de clé.

Sauf si il y a un indicateur de suppression dans le script de numérotation, il n'est pas possible de savoir si l'entrée d'utilisateur est pour un traitement KPML local ou pour d'autres receveurs du flux de supports. Donc, en l'absence d'un indicateur de suppression, l'interface d'utilisateur transmet l'entrée d'utilisateur à l'extrémité distante en temps réel, en utilisant la RFC 2833, ou en générant les tonalités appropriées, ou les deux.

3.6 DRegex

3.6.1 Généralités

Ce paragraphe est de nature informative.

La syntaxe de l'expression chiffrée régulière (Dregex, *Digit REGular EXpression*) est une transposition orientée vers la téléphonie des extensions régulières étendues (ERE, *Extended Regular Expression*) [POSIX].

KPML n'utilise pas complètement les ERE POSIX pour les raisons suivantes :

- KPML va souvent fonctionner sur des appareils à haute densité ou extrêmement faible puissance et mémoire.

- Les conventions d'application de téléphonie utilisent le symbole étoile ("*") pour la touche étoile et "x" pour tout chiffre de 0 à 9. Exiger du développeur qu'il échappe l'étoile ("*") et étende le "x" ("[0-9]") est enclin à l'erreur. Cela conduit aussi DRegex à utiliser le point (".") pour indique la répétition, qui était la fonction de l'étoile sans ornement dans ERE POSIX.

- L'expérience de mise en œuvre avec MGCP [RFC3435] et H.248.1 [RFC3525] a été que les utilisateurs et développeurs ont eu des difficultés à comprendre la préséance de l'opérateur d'alternance ("|"). Ceci est dû à la fois à la sous spécification de l'opérateur dans ces documents et de problèmes conceptuels pour les utilisateurs. Donc, le groupe de travail SIPPING a conclu que DRegex ne devrait pas prendre en charge l'alternance. Ceci dit, chaque élément de <pattern> KPML peut contenir plusieurs expressions régulières (éléments <regex>). Donc, il est normal d'avoir des schémas de remplacement (utilisation de plusieurs éléments <regex>) sans les problèmes associés à l'opérateur d'alternance ("|"). Donc, DRegex ne prend pas en charge l'opérateur d'alternance POSIX.

- DRegex inclut des classes de caractères (caractères inclus entre des crochets angulaires). Cependant, l'opérateur de négation à l'intérieur d'une classe de caractères n'opère que sur les nombres. C'est-à-dire, une classe de négation inclut implicitement A-D, *, et #. Inclure A-D, *, et # dans un opérateur de négation est un no-op. Ceux qui sont familiers de POSIX vont s'attendre à ce que la négation des chiffres 4 et 5 (par exemple, "[^45]") inclue tous les autres caractères (incluant A-D, R, *, et #) alors que ceux qui sont familiers avec les scripts de numérotation de la téléphonie vont s'attendre à ce que la négation exclue implicitement les caractères non chiffres. Comme le jeu de caractères complet de DRegex est très petit, construire une classe de négation utilisant A-D, R, *, et # exige que l'utilisateur spécifie la transposition positive inverse. Par exemple, pour spécifier toutes les touches de clés, incluant A-D et *, sauf #, la spécification serait "[0-9A-D*]" au lieu de "[^#R]".

Le tableau suivant montre la transposition de DRegex à ERE POSIX :

Dregex	ERE POSIX
*	*
.	*
x	[0-9]
[xc]	[0-9c]

Tableau 1 : Transposition de DRegex en ERE POSIX

La première substitution, qui remplace une étoile par une étoile échappée, est parce que les concepteurs d'application de téléphonie ont l'habitude d'utiliser l'étoile pour la touche étoile (très courante). Exiger une séquence d'échappement pour ce schéma très courant serait très enclin à l'erreur. De plus, l'usage de DRegex est le même que dans MGCP [RFC3435] et H.248.1 [RFC3525].
De même, l'utilisation du point à la place de l'étoile est d'un usage courant dans MGCP et H.248.1, et utiliser l'étoile dans ce contexte serait aussi une source de confusion et d'erreurs.

Le caractère "x" est un indicateur courant des chiffres de 0 à 9. On l'utilise ici, continuant la convention. Il est clair que pour le cas "[xc]", où c est tout caractère, la substitution n'est pas un remplacement aveugle de "[0-9]" pour "x", car il en résulterait "[[0-9]c]", qui n'est pas une ERE POSIX légale. La substitution pour "[xc]" est plutôt "[0-9c]".

Note : "x" n'inclut pas les caractères *, #, R, ou A à D.

Les utilisateurs doivent veiller à ne pas confondre la syntaxe DRegex avec celle des ERE POSIX. Elles ne sont PAS identiques. En particulier, il y a de nombreuses caractéristiques des ERE POSIX que DRegex ne prend pas en charge.

Comme note de mise en œuvre, si on fait les substitutions décrites dans le tableau ci-dessus, un moteur ERE POSIX standard peut analyser les chaînes de chiffres. Cependant, la transposition ne fonctionne pas en sens inverse (de ERE POSIX à DRegex). DRegex met seulement en œuvre le comportement normatif décrit ci-dessous.

3.6.2 Fonctionnement

Les espaces sont retirées avant l'analyse de DRegex. Cela permet une impression très raisonnée en XML sans affecter la signification de la chaîne DRegex.

Les règles suivantes montrent l'utilisation de DRegex dans KPML.

Entité	**Correspondance**
c	Chiffres 0 à 9, *, #, R, et A-D (insensibles à la casse)
*	Caractère *
#	Caractère #
R	Touche R (enregistrer le rappel)
[c]	Tout caractère dans le sélecteur
[^d]	Tout chiffre (0 à 9) qui n'est pas dans le sélecteur
[r1-r2]	Tout caractère dans la gamme der1 à r2, inclus
x	Tout chiffre de 0 à 9
{m}	m répétitions du schéma précédent
{m,}	m ou plus répétitions du schéma précédent
{,n}	Au plus n (incluant zéro) répétitions du schéma précédent
{m,n}	Au moins m et au plus n répétitions du schéma précédent
Lc	Correspond au caractère c si il est "long" ; c est un chiffre de 0 à 9 et A-D, #, ou *.

Entités DRegex

Pour les gammes, les caractères A-D sont disjoints des caractères 0 à 9. Si l'appareil n'a pas de touche "R", l'appareil PEUT rapporter une touche de rappel d'enregistreur comme un caractère R.

Exemple	**Description**
1	correspond au chiffre 1
[179]	correspond à 1, 7, ou 9
[2-9]	correspond à 2, 3, 4, 5, 6, 7, 8, 9
[^15]	correspond à 0, 2, 3, 4, 6, 7, 8, 9
[02-46-9A-D]	correspond à 0, 2, 3, 4, 6, 7, 8, 9, A, B, C, D
x	correspond à 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
*6[179#]	correspond à *61, *67, *69, ou *6#
x{10}	dix chiffres (0 à 9)
011x{7,15}	011 suivi par sept chiffres parmi quinze
L*	étoile longue

Exemples DRegex

3.7. Direction de surveillance

SIP identifie les dialogues par leur identifiant de dialogue. L'identifiant de dialogue est une entité d'étiquette distante, une entité d'étiquette locale, et un Call-ID, définis dans la [RFC3261].

Une méthode pour déterminer l'identifiant de dialogue, en particulier pour les applications tierces, est le paquetage de dialogue SIP de la [RFC4235].

Pour la plupart des situations, comme un appel monaural en point à point avec un seul codec, le flux à surveiller est évident. Dans de telles situations l'application n'a pas besoin de spécifier quel flux surveiller.

Mais il peut y avoir une ambiguïté à spécifier seulement le dialogue SIP à surveiller. Le dialogue peut spécifier plusieurs flux SDP qui pourraient porter des événements de pression de touches. Par exemple, un dialogue peut avoir plusieurs flux audios. Chaque fois que possible, l'interface d'utilisateur PEUT applique une politique locale pour lever l'ambiguïté sur le ou les flux à surveiller. Afin d'avoir un mécanisme extensible pour identifier les flux, le mécanisme pour spécifier les flux est comme un contenu d'élément de l'étiquette <stream>. Le seul contenu défini aujourd'hui est l'étiquette <stream>reverse</stream>.

Par défaut, l'interface d'utilisateur surveille les frappes de clés émanant de l'interface d'utilisateur. Étant donnés un identifiant de dialogue de Call-ID, une étiquette locale, et une étiquette distante, l'interface d'utilisateur surveille les frappes de clés associées à l'étiquette locale.

Dans le cas d'un support avec mandataire, et potentiellement d'autres cas, il est nécessaire de surveiller les frappes de touches qui arrivent de l'agent d'utilisateur distant. L'élément facultatif <stream> de l'étiquette <request> spécifie quel flux surveiller. La seule valeur légale est "reverse", qui signifie de surveiller le flux associé à l'étiquette distante. L'interface d'utilisateur DOIT ignorer les autres valeurs.

Note : La raison en est que l'étiquette est telle que le choix du flux individuel, si nécessaire, peut être traité de façon rétro compatible. Une spécification plus poussée du flux à surveiller est l'objet d'une future norme.

3.8 Abonnements simultanés multiples

Une application PEUT enregistrer plusieurs schémas d'entrée d'utilisateur dans un seul abonnement KPML. Si l'interface d'utilisateur supporte plusieurs abonnements KPML simultanés, l'application installe les abonnements soit dans de nouveaux dialogues initiés par SUBSCRIBE, soit dans un dialogue initié par SUBSCRIBE existant avec une nouvelle étiquette d'identifiant d'événement. Si l'interface d'utilisateur ne supporte pas les abonnements KPML simultanés, l'interface d'utilisateur DOIT répondre avec un code d'état KPML approprié.

Certaines interfaces d'utilisateur peuvent prendre en charge plusieurs abonnements de notification d'événement de frappe de touches en même temps. Dans cette situation, l'interface d'utilisateur honore chaque abonnement individuellement et indépendamment.

Un agent d'utilisateur SIP peut demander plusieurs abonnements sur le même dialogue SUBSCRIBE, en utilisant le paramètre id sur la demande d'événement kpml.

Un ou plusieurs agents d'utilisateur SIP peuvent demander des abonnements indépendants sur des dialogues SIP différents, mais réutiliser le même dialogue pour plusieurs abonnements N'EST PAS RECOMMENDÉ.

Si l'interface d'utilisateur ne prend pas en charge les abonnements multiples simultanés, l'interface d'utilisateur DOIT retourner un document KPML avec le code d'état KPML réglé à 533 sur le dialogue qui a demandé le second abonnement. L'interface d'utilisateur NE DOIT PAS modifier l'état du premier abonnement en prenant en compte la tentative de second abonnement.

4. Définition formelle de paquetage d'événement

4.1 Nom de paquetage d'événement

Le présent document définit un paquetage d'événements SIP comme défini dans la [RFC3265]. Le nom du jeton du paquetage d'événement pour ce paquetage est "kpml".

4.2. Paramètres de paquetage d'événement

Ce paquetage définit trois paramètres de paquetage d'événement : call-id, remote-tag, et local-tag. Ces paramètres DOIVENT être présents, pour identifier le dialogue d'abonnement. L'interface d'utilisateur confronte l'étiquette locale à l'étiquette to, l'étiquette distante à l'étiquette from, et l'identifiant d'appel au Call-ID.

L'ABNF pour ces paramètres est ci-dessous. Il se réfère aux nombreuses constructions de l'ABNF de la RFC 3261, comme EQUAL, DQUOTE, et token.

```
call-id  =  "call-id" EQUAL ( token / DQUOTE callid DQUOTE )
;; Note : tout DQUOTE à l'intérieur de callid DOIT être échappé !
remote-tag  =  "remote-tag" EQUAL token
local-tag  =  "local-tag" EQUAL token
```

Si un call-id contient des guillemets incorporés, ces guillemets DOIVENT être échappés en utilisant le mécanisme de citation avec barre oblique inverse (\). Noter que le paramètre call-id peut devoir être exprimé comme une chaîne entre guillemets. C'est parce que l'ABNF pour la production callid et la production de mot, qui est utilisé par callid (toutes deux de la [RFC3261]) permet certains caractères (comme "@", "[", et ":") qui ne sont pas permis dans un jeton.

4.3 Corps SUBSCRIBE

Les applications qui utilisent ce paquetage d'événements incluent un corps application/ kpml-request+xml dans les demandes SUBSCRIBE pour indiquer quels schémas de chiffres les intéressent. La syntaxe de ce type de corps est formellement décrite au paragraphe 5.2.

4.4 Durée d'abonnement

La durée de vie de l'abonnement devrait être plus longue que la durée d'appel attendue. Les abonnements à ce paquetage d'événements PEUVENT aller de quelques minutes à des semaines. Les abonnement en heures ou jours sont plus typiques et sont RECOMMENDÉS. La durée d'abonnement par défaut pour ce paquetage d'événements est 7200 secondes.

Les abonnés DOIVENT être capables de traiter une interface d'utilisateur qui retourne une valeur de Expires plus petite que la valeur demandée. Selon la [RFC3265], la durée de l'abonnement est la valeur retournée par le notificateur dans l'en-tête 200 OK Expires.

4.5 Corps NOTIFY

Les demandes NOTIFY peuvent contenir des corps application/kpml-response+xml (réponse KPML). La syntaxe de ce type de corps est formellement décrite au paragraphe 5.3. Les demandes NOTIFY dans une réponse immédiate à une demande SUBSCRIBE NE DOIVENT PAS contenir de corps sauf si elles notifient à l'abonné une condition d'erreur ou des chiffres précédemment mis en mémoire tampon.

Les notificateurs PEUVENT envoyer des notifications dans tout format acceptable à l'abonné (sur la base de l'inclusion par l'abonné de ces formats dans un en-tête Accept). Une future extension POURRA définir d'autres corps NOTIFY. Si il n'y a pas d'en-tête "Accept" dans le SUBSCRIBE, le type de corps défini dans le présent document DOIT être supposé.

4.6 Génération par l'abonné des demandes SUBSCRIBE

Un document de demande kpml contient un élément <pattern> avec une série d'étiquettes <regex>. Chaque élément <regex> spécifie un schéma potentiel auquel l'interface d'utilisateur doit correspondre. Le paragraphe 5.1 décrit le langage DRegex, ou expression numérique régulière. KPML spécifie les filtres de notification d'événement de frappe de clés. Le

type MIME pour les demandes KPML est application/kpml-request+xml.

Le document de demande KPML DOIT être bien formé et DEVRAIT être valide. Les documents KPML DOIVENT se conformer à XML 1.0 [XML2] et DOIVENT utiliser le codage UTF-8.

À cause de la nature potentiellement sensible des informations rapportées par KPML, les abonnés DEVRAIENT utiliser sips: et PEUVENT utiliser S/MIME sur le contenu.

Les abonnés DOIVENT être prêts à ce que le notificateur insiste sur l'authentification de la demande d'abonnement. Les abonnés DOIVENT être prêts à ce que le notificateur insiste pour l'utilisation d'un canal de communication sûr.

4.7 Traitement par le notificateur des demandes SUBSCRIBE

Les informations d'utilisateur transportées par KPML sont potentiellement sensibles. Par exemple, elles pourraient inclure un numéro de carte d'appel ou de crédit. Donc, l'interface d'utilisateur (le notificateur) DOIT authentifier la partie demandeuse d'une façon ou d'une autre avant d'accepter l'abonnement.

Les interfaces d'utilisateur DOIVENT mettre en œuvre l'authentification par résumé de SIP comme exigé par la [RFC3261] et DOIVENT mettre en œuvre le schéma sips: et TLS.

À l'authentification du demandeur, l'interface d'utilisateur détermine si celui-ci a l'autorisation de surveiller les frappes de clés de l'utilisateur. La politique d'autorisation par défaut est de permettre à un abonné KPML qui peut s'authentifier avec une identité spécifique de surveiller les frappes de clés à partir des sessions SIP dans lesquelles la même identité authentifiée ou son équivalent est un participant. De plus, KPML va souvent être utilisé, par exemple, entre des "serveurs d'application" (abonnés) et des passerelles du RTPC (notificateurs) gérés par le même domaine ou fédérations de domaines. Dans cette situation, un notificateur PEUT être configuré avec une liste d'abonnés qui sont spécifiquement de confiance et autorisés à souscrire aux informations de frappe de clés relatives à toutes les sessions dans un contexte particulier.

L'interface d'utilisateur retourne un URI Contact qui peut avoir des propriétés d'URI d'agent d'utilisateur mondialement acheminable *(*GRUU, *Globally Routable User Agent URI)* [RFC5627] dans l'en-tête Contact d'une réponse SIP INVITE, 1xx, ou 2xx.

Après avoir autorisé la demande, l'interface d'utilisateur vérifie si la demande est pour terminer un abonnement. Si la demande va terminer l'abonnement, l'interface d'utilisateur fait le traitement approprié, incluant les procédures décrites au paragraphe 5.2.

Si la demande n'a pas de corps KPML, alors tout document KPML courant sur ce dialogue et visé par l'identifiant d'événement, si il est présent, se termine immédiatement. C'est un mécanisme pour décharger un document KPML tout en gardant actif le dialogue initié par le SUBSCRIBE. Ceci peut être important pour des sessions sécurisées qui ont un fort coût d'établissement de session. L'interface d'utilisateur suit les procédures décrites au paragraphe 5.2.

Si le dialogue référencé par l'abonnement kpml n'existe pas, l'interface d'utilisateur suit les procédures du paragraphe 5.3. Noter que l'interface d'utilisateur DOIT produire un 200 OK à la demande SUBSCRIBE avant de produire le NOTIFY, car le SUBSCRIBE lui-même est bien formé.

Si la demande a un corps KPML, l'interface d'utilisateur analyse le document KPML. L'interface d'utilisateur DEVRAIT valider le document XML par rapport au schéma présenté au paragraphe 5.2. Si le document n'est pas valide, l'interface d'utilisateur rejette la demande SUBSCRIBE avec une réponse d'erreur appropriée et termine l'abonnement. si il y a un document KPML chargé sur l'abonnement, l'interface d'utilisateur décharge le document.

De plus, si il y a un document KPML chargé sur l'abonnement, l'appareil d'extrémité décharge le document.

Suivant la sémantique de SUBSCRIBE, si l'interface d'utilisateur reçoit un réabonnement, l'interface d'utilisateur DOIT terminer la demande KPML existante et la remplacer par la nouvelle demande.

Il est possible pour l'usage INVITE du dialogue de se terminer durant la collecte des frappes de clés. Les cas énumérés ici sont la terminaison explicite d'abonnement, la terminaison automatique d'abonnement, et la terminaison du dialogue sous-jacent (initiée par INVITE).

Si une demande SUBSCRIBE qui a un délai d'expiration de zéro (terminaison SUBSCRIBE explicite) inclut un document

KPML, et qu'il y a des entrées d'utilisateur en mémoire tampon, alors l'interface d'utilisateur tente de traiter les chiffres en mémoire tampon par rapport au document. Si il y a une correspondance, l'interface d'utilisateur DOIT générer la rapport KPML approprié avec le code d'état KPML de 200. Le corps SIP NOTIFY termine l'abonnement en réglant l'état de l'abonnement à "terminé" et une raison de "timeout".

Si la demande SUBSCRIBE a une durée de vie de zéro et pas de corps KPML ou si le temporisateur d'expiration sur le dialogue initié par le SUBSCRIBE se termine à l'interface d'utilisateur (notificateur) alors l'interface d'utilisateur DOIT produire un rapport KPML avec le code d'état KPML 487, Abonnement expiré. Le rapport inclut aussi les entrées d'utilisateur collectées jusqu'au moment de l'expiration du temporisateur ou quand l'abonnement dont la durée de vie égale à zéro a été traité. Ce peut être la chaîne nulle.

Selon les mécanismes de la [RFC3265], l'interface d'utilisateur DOIT terminer le dialogue SIP SUBSCRIBE. L'interface d'utilisateur le fait via le corps SIP NOTIFY qui transporte le rapport final décrit au paragraphe précédent. En particulier, l'état d'abonnement va être "terminated" et la raison de "timeout".
Terminer l'abonnement quand un dialogue se termine assure la réautorisation (si nécessaire) pour le rattachement aux abonnements suivants.

Si une demande SUBSCRIBE fait référence à un dialogue qui n'est pas présent à l'interface d'utilisateur, l'interface d'utilisateur DOIT générer un rapport KPML avec le code d'état KPML 481, Dialogue pas trouvé. L'interface d'utilisateur termine l'abonnement en réglant l'état d'abonnement à "terminated".

Si le document KPML n'est pas valide, l'interface d'utilisateur génère un rapport KPML avec le code d'état KPML 501, Mauvais document. L'interface d'utilisateur termine l'abonnement en réglant l'état d'abonnement à "terminated".

Si le document est valide mais si l'interface d'utilisateur ne prend pas en charge un espace de noms dans le document, l'interface d'utilisateur DOIT répondre avec un code d'état KPML 502, Espace de noms non pris en charge.

4.8 Génération par le notificateur des demandes NOTIFY

Immédiatement après l'acceptation d'un abonnement, le notificateur DOIT envoyer un NOTIFY avec les informations de localisation actuelle comme approprié sur la base de l'identité de l'abonné. Cela permet à l'abonné de resynchroniser son état.

L'interface d'utilisateur (le notificateur dans le vocabulaire SUBSCRIBE/NOTIFY) génère des demandes NOTIFY sur la base des exigences de la [RFC3265]. Précisément, si une demande SUBSCRIBE est valide et autorisée, il va en résulter un NOTIFY immédiat.

La charge utile KPML distingue entre un NOTIFY initial et un NOTIFY qui informe des frappes de touches. Si il n'y a pas d'entrée d'utilisateur en mémoire tampon au moment du SUBSCRIBE (voir ci-dessous) ou si les entrées d'utilisateur en mémoire tampon ne correspondent pas à un nouveau document KPML, alors le NOTIFY immédiat NE DOIT PAS contenir de corps KPML. Si l'interface d'utilisateur a des entrées d'utilisateur en mémoire tampon qui résultent en une correspondance utilisant le nouveau document KPML, alors le NOTIFY DOIT retourner le document KPML approprié.

Le NOTIFY en réponse à une demande SUBSCRIBE n'a pas de KPML si il n'y a pas de chiffres correspondants dans la mémoire tampon. La Figure 10 en donne un exemple.

Si il y a dans la demande SUBSCRIBE des chiffres en mémoire tampon qui correspondent à un schéma, alors le message NOTIFY en réponse à la demande SUBSCRIBE DOIT inclure le document KPML approprié.

```
NOTIFY sip:application@example.com SIP/2.0
  Via: SIP/2.0/UDP proxy.example.com
  Max-Forwards: 70
  To: <sip:application@example.com>
  From: <sip:endpoint@example.net>
  Call-Id: 439hu409h4h09903fj0ioij
  Subscription-State: active; expires=7200
  CSeq: 49851 NOTIFY
  Event: kpml
```

Figure 10 : Exemple de NOTIFY immédiat

Tous les abonnements DOIVENT être authentifiés, en particulier ceux qui correspondent sur une entrée en mémoire tampon.

KPML spécifie le format de rapport de notification de frappe de touches. Le type MIME pour les rapports KPML est application/kpml-response+xml. Le type MIME par défaut pour le paquetage d'événements kpml est application/ kpml-response+xml.

Si le demandeur n'utilise pas un protocole de transport sûr comme TLS pour chaque bond (par exemple, en utilisant un URI sips:) l'interface d'utilisateur DEVRAIT utiliser S/MIME pour protéger les informations d'utilisateur dans les réponses.

Quand l'utilisateur entre des frappes de touches qui correspondent à une étiquette <regex>, l'interface d'utilisateur va produire un rapport.

Après le rapport, l'interpréteur termine la session KPML sauf si l'abonnement a un indicateur de persistance. Si l'abonnement n'a pas d'indicateur de persistance, l'interface d'utilisateur DOIT régler l'état de l'abonnement à "terminated" dans le rapport NOTIFY.

Si l'abonnement a un indicateur de persistance, pour collecter plus de chiffres, le demandeur doit produire une nouvelle demande.

Note : Ceci souligne la nature "à un seul coup" de KPML, reflétant l'équilibre des caractéristiques et la facilité de mise en œuvre d'un interpréteur.

Les rapports KPML ont deux attributs obligatoires, code et texte. Ces attributs décrivent l'état de l'interpréteur KPML sur l'interface d'utilisateur. Noter que le code d'état KPML n'est pas nécessairement en relation avec le code de résultat SIP. Un important exemple en est lorsque une demande légale d'abonnement SIP reçoit un 200 OK SIP normal suivi par un NOTIFY, mais qu'il y a quelque chose qui ne va pas dans la demande KPML. Dans ce cas, le NOTIFY va inclure le code d'état KPML dans le rapport KPML. Noter que du point de vue de SIP, le SUBSCRIBE et le NOTIFY ont réussi. Aussi, si l'échec de KPML n'est pas récupérable, l'interface d'utilisateur va très probablement régler l'état d'abonnement à "terminated". Cela fait savoir à la machinerie SIP que l'abonnement n'est plus actif.

Si un schéma correspond, l'interface d'utilisateur va émettre un rapport KPML. Comme c'est un rapport de succès, le code est "200", et le texte est "OK".

Le rapport KPML inclut les chiffres réels qui correspondent dans l'attribut digit. La chaîne digit utilise les caractères conventionnels '*' et '#' pour respectivement étoile et dièse. Le rapport KPML inclut aussi l'attribut étiquette si la regex qui correspond aux chiffres avait un attribut étiquette.

Si l'abonnement demandait la suppression des chiffres et si l'interface d'utilisateur a supprimé les chiffres, l'attribut supprimé indique "vrai". La valeur par défaut de supprimé est "faux".

Note : KPML n'inclut pas d'horodatage. Il y a un certain nombre de raisons à cela. D'abord, quel horodatage inclure ? Serait-ce l'heure de la première frappe de touche détectée ? L'heure à laquelle l'interpréteur a collecté la chaîne entière ? Une gamme ? Ensuite, si l'horodatage RTP est un événement digne d'intérêt, pourquoi ne pas simplement mettre RTP à la première place ? Cela dit, si il est vraiment impérieux d'avoir l'horodatage dans la réponse, ce pourrait être un attribut de l'étiquette <response>.

Noter que si le dialogue (initié par INVITE) surveillé se termine, le notificateur DOIT encore terminer explicitement les abonnements KPML qui surveillent ce dialogue.

4.9 Traitement des demande NOTIFY par l'abonné

Si il n'y a pas de corps KPML, cela signifie que le SUBSCRIBE a réussi. Cela établit le dialogue si il n'y a pas d'entrée d'utilisateur en mémoire tampon à rapporter.

Si il y a un document KPML, et si le code d'état KPML est 200, une correspondance s'est alors produite.

Si il y a un document KPML, et si le code d'état KPML est entre 400 et 499, une erreur s'est alors produite avec la collecte d'entrée d'utilisateur. La cause la plus probable est une condition de fin de temporisation.

Si il y a un document KPML, et si le code d'état KPML est entre 500 et 599, une erreur s'est alors produite avec l'abonnement. Voir à la Section 6 la signification des codes d'état KPML.
L'abonné DOIT être conscient de l'état de l'abonnement. L'interface d'utilisateur peut terminer l'abonnement à tout moment.

4.10 Traitement des demandes fourchées

Les demandes fourchées NE SONT PAS PERMISES pour ce type d'événement. On peut s'en assurer si les abonnements à ce paquetage d'événements sont envoyés à des URI SIP qui ont des propriétés de GRUU.

4.11 Taux des notifications

L'interface d'utilisateur NE DOIT PAS générer des messages plus vite que 25 messages par seconde, ou un message toutes les 40 millisecondes. C'est la durée minimum pour les écoulements de chiffres MF. Même des DTMF de 30 millisecondes, comme on en trouve parfois au Japon, ont un intervalle de temps de 20 millisecondes, d'où résulte un temps entre chiffres de 50 millisecondes. Le présent document RECOMMENDE fortement de ne pas utiliser KPML pour des messages chiffre par chiffre, comme ce serait le cas si la seule <regex> était "x".

Le taux soutenu de notifications ne devra pas être de plus de 100 Notify par minute.

L'interface d'utilisateur DOIT livrer les notifications de façon fiable. Comme il n'y a pas de métrique significative pour ralentir les demandes, l'interface d'utilisateur DEVRAIT envoyer les messages NOTIFY sur un transport à encombrement contrôlé, comme TCP.
Noter que toutes les mises en œuvre de SIP sont déjà obligées de mettre en œuvre SIP sur TCP.

4.12 Agents d'état et listes

Les demandes KPML sont envoyées à un URI SIP spécifique, qui peut avoir des propriétés de GRUU, et elles tentent de surveiller un flux spécifique qui correspond à un dialogue cible spécifique. Par conséquent, les mises en œuvre NE DOIVENT PAS définir des agents d'état pour ce paquetage d'événements ou permettre des abonnements pour ce paquetage d'événements à des listes de ressources utilisant l'extension de liste d'événements [RFC4662].

4.13 Comportement d'un serveur mandataire

Il n'y a pas d'exigence supplémentaire sur les mandataires SIP, autres que de transmettre de façon transparente les méthodes SUBSCRIBE et NOTIFY comme exigé dans SIP.

5. Syntaxe formelle

5.1 DRegex

La définition suivante suit la [RFC4234]. La définition de CHIFFRE est de la RFC 4234, à savoir les caractères "0" à "9". Noter que le DRegexCharacter n'est pas un HEXDIG *(chiffre hexadécimal)* de la RFC 4234. En particulier, DRegexCharacter n'inclut ni "E" ni "F". Noter que DRegexCharacter est insensible à la casse.

```
DRegex             = 1*( DRegexPosition [ RepeatCount ] )
DRegexPosition     = DRegexSymbol / DRegexSet
DRegexSymbol       = [ "L" ] DRegexCharacter
DRegexSet          = "[" 1*DRegexSetList "]"
DRegexSetList      = DRegexCharacter [ "-" DRegexCharacter ]
DregexCharacter    = DIGIT / "A" / "B" / "C" / "D" / "R" / "X" / "*" / "#" / "a" / "b" / "c" / "d" / "r"
RepeatCount        = "." / "{" RepeatRange "}"
RepeatRange        = Count / ( Count "," Count ) / ( Count "," ) / ( "," Count )
Count              = 1*CHIFFRE
```

ABNF pour DRegex

Noter que de futures extensions au présent document pourront introduire d'autres caractères pour DRegexCharacter, dans le schéma de H.248.1 [RFC3525] ou éventuellement comme chaînes nommées ou espaces de noms XML.

5.2 Demande KPML

La syntaxe suivante pour les demandes KPML utilise le schéma XML [XML].

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:kpml-request"
 xmlns="urn:ietf:params:xml:ns:kpml-request"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">
  <xs:element name="kpml-request">
    <xs:annotation>
      <xs:documentation>Demande en langage de balisage de stimulus clavier de l'IETF
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="stream" minOccurs="0">
          <xs:complexType>
            <xs:choice>
              <xs:element name="reverse" minOccurs="0"/>
              <xs:any namespace="##other"/>
            </xs:choice>
          </xs:complexType>
        </xs:element>
        <xs:element name="pattern">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="flush" minOccurs="0">
                <xs:annotation>
                  <xs:documentation>
                      Le comportement par défaut est de ne pas purger la mémoire tampon
                  </xs:documentation>
                </xs:annotation>
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:string"/>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
              <xs:element name="regex" maxOccurs="unbounded">
                <xs:annotation>
                  <xs:documentation>
                    La notation de frappe de touches est une chaîne pour permettre une future extension de touches non de 16
                    chiffres ou de clé désignées
                  </xs:documentation>
                </xs:annotation>
                <xs:complexType mixed="true">
                  <xs:choice>
                    <xs:element name="pre" minOccurs="0">
                      <xs:complexType>
                        <xs:simpleContent>
                          <xs:extension base="xs:string"/>
                        </xs:simpleContent>
                      </xs:complexType>
                    </xs:element>
                    <xs:any namespace="##other"/>
```

```
            </xs:choice>
            <xs:attribute name="tag" type="xs:string" usage="optional"/>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
         <xs:attribute name="persist" usage="optional">
          <xs:annotation>
           <xs:documentation>Le comportement par défaut est "un seul coup"
           </xs:documentation>
          </xs:annotation>
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="un seul coup"/>
            <xs:enumeration value="persist"/>
            <xs:enumeration value="single-notify"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="interdigittimer" type="xs:integer" usage="optional">
          <xs:annotation>
           <xs:documentation>Par défaut 4000 (ms)
           </xs:documentation>
          </xs:annotation>
         </xs:attribute>
         <xs:attribute name="criticaldigittimer" type="xs:integer" usage="optional">
          <xs:annotation>
           <xs:documentation>Par défaut 1000 (ms)
            </xs:documentation>
          </xs:annotation>
         </xs:attribute>
         <xs:attribute name="extradigittimer" type="xs:integer" usage="optional">
          <xs:annotation>
           <xs:documentation>Par défaut 500 (ms)
           </xs:documentation>
          </xs:annotation>
         </xs:attribute>
         <xs:attribute name="long" type="xs:integer" usage="optional"/>
         <xs:attribute name="longrepeat" type="xs:boolean" usage="optional"/>
         <xs:attribute name="nopartial" type="xs:boolean" usage="optional">
          <xs:annotation>
           <xs:documentation>Par défaut est faux
           </xs:documentation>
          </xs:annotation>
         </xs:attribute>
         <xs:attribute name="enterkey" type="xs:string" usage="optional">
          <xs:annotation>
           <xs:documentation>Pas d'enterkey par défaut
           </xs:documentation>
          </xs:annotation>
         </xs:attribute>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" usage="required"/>
     </xs:complexType>
    </xs:element>
   </xs:schema>
```

Figure 12 : schéma XML pour demandes KPML

5.3 Réponse KPML

La syntaxe suivante pour les réponses KPML utilise le schéma XML [XML].

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:kpml-response"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="urn:ietf:params:xml:ns:kpml-response"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">
  <xs:element name="kpml-response">
    <xs:annotation>
      <xs:documentation>Réponse en langage de balisage de stimulus clavier de l'IETF
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
   <xs:attribute name="version" type="xs:string" usage="required"/>
      <xs:attribute name="code" type="xs:string" usage="required"/>
      <xs:attribute name="text" type="xs:string" usage="required"/>
      <xs:attribute name="suppressed" type="xs:boolean" usage="optional"/>
       <xs:attribute name="forced_flush" type="xs:string" usage="optional">
        <xs:annotation>
          <xs:documentation>
            Chaîne pour utilisation future, par exemple, de numéros de chiffres perdus.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="digits" type="xs:string" usage="optional"/>
      <xs:attribute name="tag" type="xs:string" usage="optional">
        <xs:annotation>
          <xs:documentation>Étiquettes correspondantes au regex dans la demande
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure 13 : schéma XML pour réponses KPML

6. Énumération des codes d'état KPML

Les codes d'état KPML suivent en gros leur équivalent SIP. Les codes qui commencent par 2 indiquent le succès. Les codes qui commencent par 4 indiquent l'échec. Les codes qui commencent par 5 indiquent une défaillance du serveur, généralement une défaillance à interpréter le document ou à prendre en charge une caractéristique demandée.

Les clients KPML DOIVENT être capables de traiter des codes d'état arbitraires en examinant seulement le premier chiffre.

Tout texte peut être dans un document de rapport KPML. Les clients KPML NE DOIVENT PAS interpréter le champ texte.

Code	Texte
200	Succès
402	Terminé par l'utilisateur sans correspondance
423	Expiration du temporisateur
481	Dialogue non trouve
487	Abonnement expiré
501	Mauvais document
502	Espace de noms non pris en charge

531	Abonnements persistants non pris en charge
532	Expressions régulières multiples non prises en charge
533	Abonnements multiples sur un dialogue non pris en charge
534	Trop d'expressions régulières

Tableau 4 : Codes d'état KPML

7. Considérations relatives à l'IANA

Le présent document enregistre un nouveau paquetage d'événements SIP, deux nouveaux types MIME, et deux nouveaux espaces de noms XML.

7.1 Enregistrement de paquetage d'événement SIP

Nom de paquetage : kpml
Type : paquetage
Contact : Eric Burger, <e.burger@ieee.org>
Contrôleur des changements : Groupe de travail SIPPING sur délégation de l'IESG
Spécification publiée : RFC 4730

7.2 Type de support MIME application/kpml-request+xml

Nom de type de support MIME : application
Nome de sous type MIME : kpml-request+xml
Paramètres exigés : aucun
Paramètre facultatif : le même que le paramètre de jeu de caractère application/xml comme spécifié dans les types de support XML [RFC3023]
Considérations de codage : voir la [RFC3023].
Considérations de sécurité : voir la Section 10 de la [RFC3023] et la Section 8 de la RFC4730.
Considérations d'interopérabilité : voir la [RFC3023] et la RFC4730.
Spécification publiée : RFC 4730
Applications qui utilisent ce type de supports : applications en mode session qui ont des interfaces primitives d'utilisateur.
Contrôleur des changements : groupe de travail SIPPING sur délégation de l'IESG.
Adresse personnelle et de messagerie pour plus d'informations: Eric Burger <e.burger@ieee.org>
Usage prévu : COMMUN

7.3 Type de support MIME application/kpml-response+xml

Nom de type de support MIME : application
Nom de sous type MIME : kpml-response+xml
Paramètres exigés : aucun
Paramètre facultatif : le même que le paramètre de jeu de caractère application/xml comme spécifié dans les types de support XML [RFC3023]
Considérations de codage : voir la [RFC3023].
Considérations de sécurité : voir la Section 10 de la [RFC3023] et la Section 8 de la RFC4730.
Considérations d'interopérabilité : voir la [RFC3023] et la RFC4730.
Spécification publiée : RFC 4730
Applications qui utilisent ce type de supports : applications en mode session qui ont des interfaces primitives d'utilisateur.
Contrôleur des changements : groupe de travail SIPPING sur délégation de l'IESG.
Adresse personnelle et de messagerie pour plus d'informations: Eric Burger <e.burger@ieee.org>
Usage prévu : COMMUN

7.4 Enregistrement de sous espace d'URN pour urn:ietf:params:xml:ns:kpml-request

URI : urn:ietf:params:xml:ns:kpml-request

Contact d'enregistrement : IESG <iesg@ietf.org>

XML :

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML Basic 1.0//EN"
      "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
   <meta http-equiv="content-type"
      content="text/html;charset=iso-8859-1"/>
   <title>Key Press Markup Language Request</title>
  </head>
  <body>
   <h1>Namespace for Key Press Markup Language Request</h1>
   <h2>urn:ietf:params:xml:ns:kpml-request</h2>
   <p>
<a href="ftp://ftp.rfc-editor.org/in-notes/RFC4730.txt">RFC 4730</a>.
   </p>
  </body>
</html>
```

7.5 Enregistrement de sous espace d'URN pour urn:ietf:params:xml:ns:kpml-response

URI : urn:ietf:params:xml:ns:kpml-response

Contact d'enregistrement : IESG <iesg@ietf.org>

XML :

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML Basic 1.0//EN"
      "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
   <meta http-equiv="content-type"
      content="text/html;charset=iso-8859-1"/>
   <title>Key Press Markup Language Response</title>
  </head>
  <body>
   <h1>Namespace for Key Press Markup Language Response</h1>
   <h2>urn:ietf:params:xml:ns:kpml-response</h2>
   <p>
<a href="ftp://ftp.rfc-editor.org/in-notes/rfc4730.txt">RFC 4730</a>.
   </p>
  </body>
</html>
```

7.6 Enregistrement de schéma de demande KPML

Selon la [RFC3688], l'IANA a enregistré le schéma XML pour KPML comme référencé au paragraphe 5.2 de la RFC 4730.

URI : urn:ietf:params:xml:schema:kpml-request

Contact d'enregistrement : <iesg@ietf.org>

7.7 Enregistrement de schéma de réponse KPML

Selon la [RFC3688], l'IANA a enregistré le schéma XML pour KPML comme référencé au paragraphe 5.3 de la RFC 4730.

URI : urn:ietf:params:xml:schema:kpml-response

Contact d'enregistrement : IETF, groupe de travail SIPPING <sipping@ietf.org>, Eric Burger <e.burger@ieee.org>.

8. Considérations sur la sécurité

Les informations d'utilisateur transportées par KPML sont potentiellement sensibles. Par exemple, elles pourraient inclure des numéros de carte d'appel ou de crédit. Ces informations potentiellement confidentielles pourraient être fournies accidentellement si le notificateur n'authentifie ou n'autorise pas de façon appropriée un abonnement. Des informations aussi confidentielles (comme un numéro de carte de crédit ou d'appel) pourraient être révélées à un autre abonné par ailleurs légitime (qui opère un IVR) si les chiffres mis plus tôt en mémoire tampon dans la session sont fournis involontairement au nouvel abonné.

De même, un espion pourrait voir les informations de chiffres K¨ML si ils ne sont pas chiffrés, ou un attaquant pourrait injecter des notifications frauduleuses sauf si les messages ou le chemin SIP sur lequel ils voyagent sont protégés en intégrité.

Donc, les interfaces d'utilisateur NE DOIVENT PAS dégrader leur propre politique de sécurité. C'est-à-dire, si la politique d'une interface d'utilisateur est de restreindre les notifications aux abonnés authentifiés et autorisés sur des communications sécurisées, alors l'interface d'utilisateur ne doit pas accepter un abonnement non authentifié, non autorisé sur un canal de communication non sûr.

En tant que balisage XML, toutes les considérations de sécurité de la [RFC3023] et de la [RFC3406] DOIVENT être satisfaites. On apportera une attention particulière aux exigences de robustesse de l'analyse XML.

Les informations de frappes de touches sont potentiellement sensibles. Par exemple, elles peuvent représenter des informations de carte de crédit, de carte d'appel, ou autres informations personnelles. La capture des sessions permet à des entités non autorisées d'accéder à ces informations sensibles. Donc, la signalisation DEVRAIT être sécurisée, par exemple, l'utilisation de TLS et sips: DEVRAIT être utilisée. De plus, l'information elle-même est sensible de sorte que S/MIME ou d'autres mécanismes appropriés DEVRAIENT être utilisés.

Les abonnements DOIVENT être authentifiés d'une manière ou d'une autre. Comme exigé par le cœur de la spécification de SIP [RFC3261], toutes les mises en œuvre de SIP DOIVENT prendre en charge l'authentification par résumé. De plus, les interfaces d'utilisateur DOIVENT mettre en œuvre la prise en charge du schéma sips: et de SIP sur TLS. Les abonnés DOIVENT s'attendre à ce que les interfaces d'utilisateur demandent l'utilisation d'un schéma d'authentification. Si la politique locale d'une interface d'utilisateur est d'utiliser l'authentification ou des canaux de communication sécurisés, l'interface d'utilisateur DOIT rejeter les demandes d'abonnement qui ne satisfont pas à cette politique.

Les interfaces d'utilisateur DOIVENT commencer à mettre en mémoire tampon les entrées d'utilisateur à réception d'un abonnement authentifié et accepté. Cette mise en mémoire tampon est faite abonnement par abonnement.

9. Exemples

Cette section est informative par nature. Si il y a une discordance entre cette section et les sections normatives ci-dessus, les sections normatives doivent l'emporter.

9.1 Surveillance des dièses

Un besoin commun des applications pré payées et des applications d'assistant personnel est de surveiller dans une conversation un signal indiquant un changement de la focalisation de l'utilisateur sur la partie appelée à travers l'application à l'application elle-même. Par exemple, si on appelle une personne en utilisant une carte d'appel pré payée, et si la personne appelée redirige l'appel sur une messagerie vocale, les chiffres frappés sont pour le système de messagerie vocale. Cependant, de nombreuses applications ont une séquence de touches spéciale, comme le dièse (#, ou signe Livre) ou *9, qui termine la session appelée et décale l'attention de l'utilisateur sur l'application.

La Figure 16 montre le KPML pour un long dièse.

```
<?xml version="1.0"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
     version="1.0">
  <pattern>
    <regex>L#</regex>
  </pattern>
</kpml-request>
```

Figure 16 : Exemple de dièse long

La valeur regex L indique que les chiffres qui suivent doivent être une pression de touche de longue durée.

9.2 Collection de chaîne de numérotation

Dans cet exemple, l'interface d'utilisateur collecte une chaîne de numérotation. L'application utilise KPML pour déterminer rapidement quand l'utilisateur entre un numéro cible. De plus, KPML indique quel type de numéro l'utilisateur a entré.

```
<?xml version="1.0"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
     version="1.0">
  <pattern>
    <regex tag="local-operator">0</regex>
    <regex tag="ld-operator">00</regex>
    <regex tag="vpn">7[x][x][x]</regex>
    <regex tag="local-number7">9xxxxxxx</regex>
    <regex tag="RI-number">9401xxxxxxx</regex>
    <regex tag="local-number10">9xxxxxxxxxx</regex>
    <regex tag="ddd">91xxxxxxxxx</regex>
    <regex tag="iddd">011x.</regex>
  </pattern>
</kpml-request>
```

Figure 17 : Exemple de code de chaîne de numérotation KPML

Noter l'utilisation de l'attribut "tag" pour indiquer quelle regex correspond à la chaîne numérotée. Le cas intéressant ici est si l'utilisateur a entré "94015551212". Cette chaîne correspond à la fois à l'expression régulière "9401xxxxxxx" et à "9xxxxxxxxxx". Les deux expressions ont la même longueur. Donc, l'interpréteur KPML va prendre la chaîne "9401xxxxxxx", car elle se produit en premier dans l'ordre du document. La Figure 18 montre la réponse.

```
<?xml version="1.0"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-resposne"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
     version="1.0"
     code="200" text="OK"
     digits="94015551212" tag="RI-number"/>
```

Figure 18 : Réponse de chaîne de numérotation KPML

10. Exemples de flux d'appels

10.1 Chiffres supplémentaires

Cette section donne un exemple non normatif d'une application qui collecte des chiffres supplémentaires. La collecte de chiffres supplémentaires se fait lorsque le réseau demande des chiffres supplémentaires après l'entrée de l'adresse de destination par l'appelant. Une chaîne de numérotation supplémentaire typique est longue de quatre chiffres.

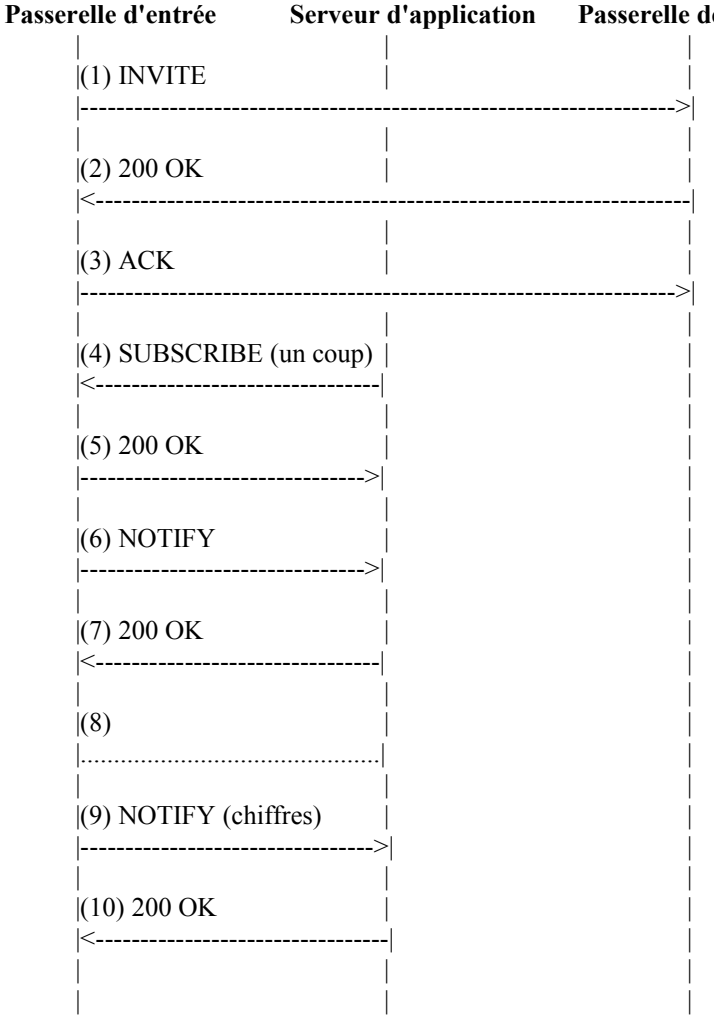


Figure 19 : Flux d'appel de chiffres supplémentaires

Dans les messages (1 à 3), la passerelle d'entrée établit un dialogue avec une passerelle de sortie. L'application apprend l'identifiant de dialogue par des mécanismes hors bande, tels que le paquetage de dialogue ou en étant co-résidente avec la passerelle de sortie. Une partie du message d'accusé de réception est donnée ci-dessous, pour illustrer les identifiants de dialogue.

```
ACK sip:gw@subA.example.com SIP/2.0
Via: ...
Max-Forwards: ...
Route: ...
From: <sip:phn@example.com>;tag=jfh21
To: <sip:gw@subA.example.com>;tag=onjwe2
Call-ID: 12345592@subA.example.com
 ...
```

Dans le message (4), l'application demande que la passerelle collecte une chaîne de quatre frappes de touches.

```
SUBSCRIBE sip:gw@subA.example.com SIP/2.0
Via: SIP/2.0/TCP client.subB.example.com;branch=q4i9ufr4ui3
From: <sip:ap@subB.example.com>;tag=567890
To: <sip:gw@subA.example.com>
Call-ID: 12345601@subA.example.com
CSeq: 1 SUBSCRIBE
Contact: <sip:ap@client.subB.example.com>
Max-Forwards: 70
Event: kpml ;remote-tag="sip:phn@example.com;tag=jfh21"
          ;local-tag="sip:gw@subA.example.com;tag=onjwe2"
          ;call-id="12345592@subA.example.com"
Expires: 7200
Accept: application/kpml-response+xml
Content-Type: application/kpml-request+xml
Content-Length: 292

<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation=
     "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
   version="1.0">
  <pattern persist="one-shot">
   <regex>xxxx</regex>
  </pattern>
</kpml-request>
```

Le message (5) est l'accusé de réception de la demande d'abonnement.

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP subB.example.com;branch=q4i9ufr4ui3;
    received=192.168.125.12
From: <sip:ap@subB.example.com>;tag=567890
To: <sip:gw@subA.example.com>;tag=1234567
Call-ID: 12345601@subA.example.com
CSeq: 1 SUBSCRIBE
Contact: <sip:gw27@subA.example.com>
Expires: 3600
Event: kpml
```

Le message (6) est la notification immédiate de l'abonnement.

```
NOTIFY sip:ap@client.subB.example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=gw27id4993
To: <sip:ap@subB.example.com>;tag=567890
From: <sip:gw@subA.example.com>;tag=1234567
Call-ID: 12345601@subA.example.com
CSeq: 1000 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3599
Max-Forwards: 70
Content-Length: 0
```

Le message (7) est l'accusé de réception du message de notification.

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP subA.example.com;branch=gw27id4993
To: <sip:ap@subB.example.com>;tag=567890
From: <sip:gw@subA.example.com>;tag=1234567
Call-ID: 12345601@subA.example.com
```

CSeq: 1000 NOTIFY

Un certain temps s'écoule (8).

L'usager entre les données. L'appareil fournit la notification des chiffres collectés dans le message (9). Comme c'était un abonnement pour un seul coup, on note que l'état d'abonnement est "terminated".

```
NOTIFY sip:ap@client.subB.example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=gw27id4993
To: <sip:ap@subB.example.com>;tag=567890
From: <sip:gw@subA.example.com>;tag=1234567
Call-ID: 12345601@subA.example.com
CSeq: 1001 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: terminated
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 258

<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
      "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
    version="1.0"
    code="200" text="OK"
    digits="4336"/>
```

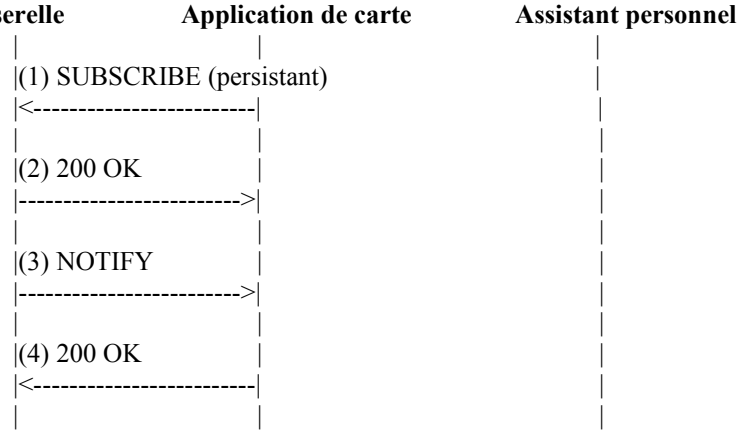
Le message (10) est l'accusé de réception de la notification.

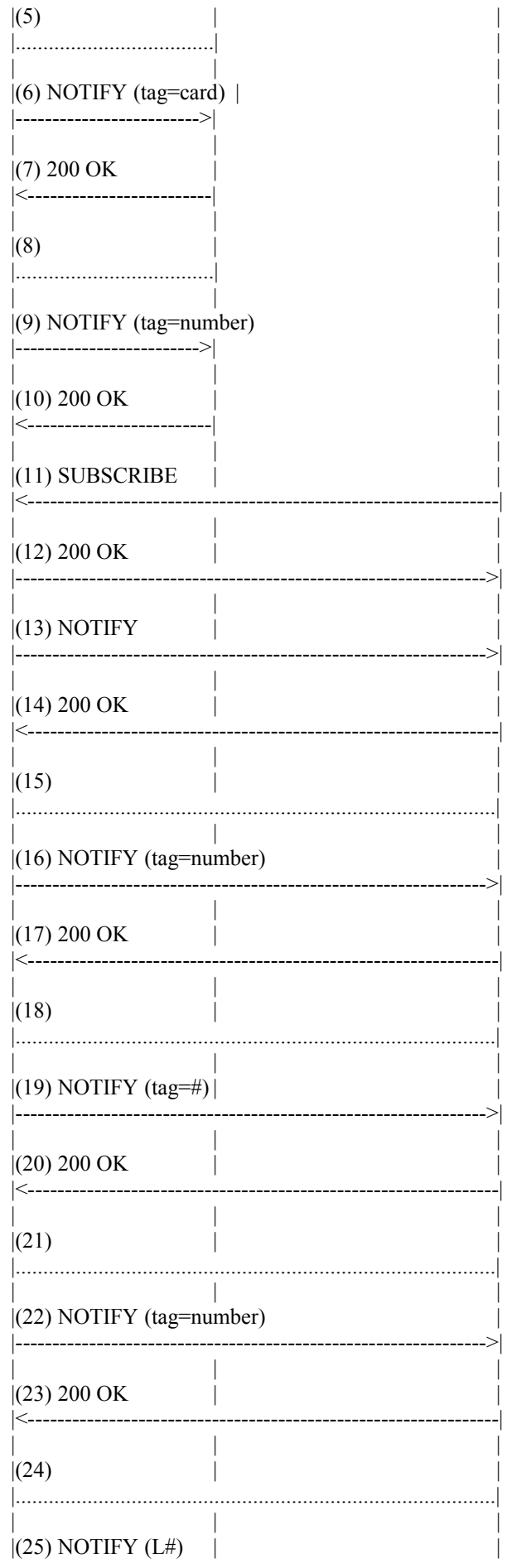
```
SIP/2.0 200 OK
Via: SIP/2.0/TCP subA.example.com;branch=gw27id4993
To: <sip:ap@subB.example.com>;tag=567890
From: <sip:gw@subA.example.com>;tag=1234567
Call-ID: 12345601@subA.example.com
CSeq: 1001 NOTIFY
```

10.2 Applications multiples

Ce paragraphe donne un exemple non normatif d'applications multiples. Une application collecte un numéro de destination à appeler. Cette application attend ensuite un "long dièse". Durant l'appel, l'appel passe à une application d'assistant personnel, qui interagit avec l'utilisateur. De plus, l'application d'assistant personnel cherche un "dièse court".

Pour la clarté de l'exposé, on ne montre pas le dialogue INVITE.




(5)
(6) NOTIFY (tag=card)
(7) 200 OK
(8)
(9) NOTIFY (tag=number)
(10) 200 OK
(11) SUBSCRIBE
(12) 200 OK
(13) NOTIFY
(14) 200 OK
(15)
(16) NOTIFY (tag=number)
(17) 200 OK
(18)
(19) NOTIFY (tag=#)
(20) 200 OK
(21)
(22) NOTIFY (tag=number)
(23) 200 OK
(24)
(25) NOTIFY (L#)

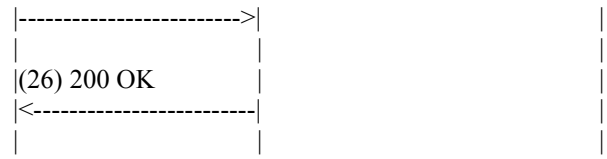


Figure 27 : Flux d'appel d'applications multiples

Le message (1) est la demande d'abonnement pour le numéro de carte.

```
SUBSCRIBE sip:gw@subA.example.com SIP/2.0
Via: SIP/2.0/TCP client.subB.example.com;branch=3qo3j0ouq
From: <sip:ap@subB.example.com>;tag=978675
To: <sip:gw@subA.example.com>
Call-ID: 12345601@subA.example.com
CSeq: 20 SUBSCRIBE
Contact: <sip:ap@client.subB.example.com>
Max-Forwards: 70
Event: kpml ;remote-tag="<sip:phn@example.com;tag=jfi23>"
          ;local-tag="sip:gw@subA.example.com;tag=oi43jfq"
          ;call-id="12345598@subA.example.com"
Expires: 7200
Accept: application/kpml-response+xml
Content-Type: application/kpml-request+xml
Content-Length: 339

<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
     version="1.0">
  <pattern persist="persist">
    <regex tag="card">x{16}</regex>
    <regex tag="number">x{10}</regex>
  </pattern>
</kpml-request>
```

Les messages (2 à 4) ne sont pas montrés, pour faire court. Le message (6) est la notification du numéro de carte.

```
NOTIFY sip:ap@client.subB.example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=3qo3j0ouq
To: <sip:ap@subB.example.com>;tag=978675
From: <sip:gw@subA.example.com>;tag=9783453
Call-ID: 12345601@subA.example.com
CSeq: 3001 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3442
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 271

<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
     version="1.0"
     code="200" text="OK"
     digits="9999888877776666"/> tag="card"/>
```

Le message (7) est l'accusé de réception de la notification. Le temps passe en (8). Le message (9) est la notification du numéro composé.

```
NOTIFY sip:ap@client.subB.example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=3qo3j0ouq
To: <sip:ap@subB.example.com>;tag=978675
From: <sip:gw@subA.example.com>;tag=9783453
Call-ID: 12345601@subA.example.com
CSeq: 3001 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3542
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 278

<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
      "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
    version="1.0"
    code="200" text="OK"
    digits="2225551212" tag="number"/>
```

Le message (11) est la demande de surveillance d'un dièse long.

```
SUBSCRIBE sip:gw@subA.example.com SIP/2.0
Via: SIP/2.0/TCP client.subB.example.com;branch=3qo3j0ouq
From: <sip:ap@subB.example.com>;tag=978675
To: <sip:gw@subA.example.com>
Call-ID: 12345601@subA.example.com
CSeq: 21 SUBSCRIBE
Contact: <sip:ap@client.subB.example.com>
Max-Forwards: 70
Event: kpml ;remote-tag="<sip:phn@example.com;tag=jfi23>"
          ;local-tag="sip:gw@subA.example.com;tag=oi43jfq"
          ;call-id="12345598@subA.example.com"
Expires: 7200
Accept: application/kpml-response+xml
Content-Type: application/kpml-request+xml
Content-Length: 295

<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
      "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
    version="1.0">
  <pattern persist="single-notify">
    <regex>L#</regex>
  </pattern>
</kpml-request>
```

Le message (13) est la demande formulé par l'application d'assistant personnel de la surveillance du numéro et et du signe dièse.

```
SUBSCRIBE sip:gw@subA.example.com SIP/2.0
Via: SIP/2.0/TCP pahost.example.com;branch=xzvsadf
From: <sip:pa@example.com>;tag=4rgj0f
To: <sip:gw@subA.example.com>
```

```
Call-ID: 93845@pahost.example.com
CSeq: 21 SUBSCRIBE
Contact: <sip:pa12@pahost.example.com>
Max-Forwards: 70
Event: kpml ;remote-tag="<sip:phn@example.com;tag=jfi23>"
          ;local-tag="sip:gw@subA.example.com;tag=oi43jfq"
          ;call-id="12345598@subA.example.com"
Expires: 7200
Accept: application/kpml-response+xml
Content-Type: application/kpml-request+xml
Content-Length: 332

<?xml version="1.0" encoding="UTF-8"?>
<kpml-request xmlns="urn:ietf:params:xml:ns:kpml-request"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-request kpml-request.xsd"
     version="1.0">
  <pattern persist="persist">
    <regex tag="number">x{10}</regex>
    <regex tag="#">#</regex>
  </pattern>
</kpml-request>
```

Le message (18) est la notification du numéro collecté.

```
NOTIFY sip:pa@example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=xzvsadf
To: <sip:pa@example.com>;tag=4rgj0f
From: <sip:gw@subA.example.com>;tag=9788823
Call-ID: 93845@pahost.example.com
CSeq: 3021 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3540
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 278

<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation=
       "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
     version="1.0"
     code="200" text="OK" digits="3335551212" tag="number"/>
```

Le message (21) est la notification de la détection du signe dièse.

```
NOTIFY sip:pa@example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=xzvsadf
To: <sip:pa@example.com>;tag=4rgj0f
From: <sip:gw@subA.example.com>;tag=9788823
Call-ID: 93845@pahost.example.com
CSeq: 3022 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3540
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 264
```

```
<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
      "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
    version="1.0"
    code="200" text="OK"
    digits="#" tag="#"/>
```

Le message (27) est la notification du long dièse à l'application de carte.

```
NOTIFY sip:ap@client.subB.example.com SIP/2.0
Via: SIP/2.0/UDP subA.example.com;branch=3qo3j0ouq
To: <sip:ap@subB.example.com>;tag=978675
From: <sip:gw@subA.example.com>;tag=9783453
Call-ID: 12345601@subA.example.com
CSeq: 3037 NOTIFY
Contact: <sip:gw27@subA.example.com>
Event: kpml
Subscription-State: active;expires=3216
Max-Forwards: 70
Content-Type: application/kpml-response+xml
Content-Length: 256

<?xml version="1.0" encoding="UTF-8"?>
<kpml-response xmlns="urn:ietf:params:xml:ns:kpml-response"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
      "urn:ietf:params:xml:ns:kpml-response kpml-response.xsd"
    version="1.0"
    code="200" text="OK"
    digits="#"/>
```

11. Références

11.1 Références normatives

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC3023] M. Murata, S. St.Laurent et D. Kohn, "Types de support XML", janvier 2001. (*Obsolète, voir* RFC7303)

[RFC3261] J. Rosenberg et autres, "SIP : Protocole d'initialisation de session", juin 2002. (*Mise à jour par* 3265, 3853, 4320, 4916, 5393, 6665, 8217, 8760)

[RFC3265] A.B. Roach, "Notification d'événement spécifique du protocole d'initialisation de session (SIP)", juin 2002. (*MàJ par* RFC6446) (*Remplacée par la* RFC6665)

[RFC3406] L. Daigle, D. van Gulik, R. Iannella, P. Faltstrom, "Mécanismes de définition d'espace de noms de noms de ressource uniforme (URN)", octobre 2002. (BCP0066 *; rendu obsolète par* RFC8141)

[RFC3688] M. Mealling, "Registre XML de l'IETF", BCP 81, janvier 2004.

[RFC4234] D. Crocker et P. Overell, "BNF augmenté pour les spécifications de syntaxe : ABNF", octobre 2005. *(Remplace* RFC2234, *remplacée par* RFC5234)

[XML] Thompson, H., Beech, D., Maloney, M., and N. Mendelsohn, "XML Schema Part 1: Structures", W3C REC REC-xmlschema-1-20010502, mai 2001.

11.2 Références pour information

[POSIX] Institute of Electrical and Electronics Engineers, "Information Technology - Portable Operating System Interface (POSIX) - Part 1: Base Definitions, Chapter 9", IEEE Standard 1003.1, juin 2001.

[RFC2833] H. Schulzrinne, S. Petrack, "Charge utile RTP pour chiffres DTMF, tonalités téléphoniques et signaux téléphoniques", mai 2000. (*Obsolète, voir* RFC4733, RFC4734) *(P.S.)*

[RFC3435] F. Andreasen, B. Foster, "Protocole de contrôle de passerelle de support (MGCP) version 1.0", janvier 2003. (*MàJ par* RFC3661) *(Information)*

[RFC3525] C. Groves et autres, "Protocole de commande des routeurs, version 1", juin 2003. (*Obsolète, voir* RFC5125) *(Historique)*

[RFC4235] J. Rosenberg et autres, "Paquetage d'événement de dialogue initié par INVITE pour le protocole d'initialisation de session (SIP)", novembre 2005. *(P.S.)*

[RFC4662] A. B. Roach et autres, "Extension de notification d'événement du protocole d'initialisation de session (SIP) pour les listes de ressources", août 2006. *(P.S.)*

[RFC4722] J. Van Dyke et autres, "Langage de balisage de commande de serveur de supports (MSCML) et son protocole", novembre 2006. (*Obsolète, voir* RFC5022) *(Information)*

[RFC5627] J. Rosenberg, "Obtention et utilisation des URI d'agent d'utilisateur mondialement acheminable (GRUU) dans le protocole d'initialisation de session (SIP)", octobre 2009. *(P. S.)*

[RFC5629] J. Rosenberg, "Cadre de l'interaction d'application dans le protocole d'initialisation de session (SIP)", octobre 2009. *(P.S.)*

[XML2] Bray, T., Paoli, J., Sperberg-McQueen, C., and E. Maler, "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C REC REC-xml-20001006, octobre 2000.

Appendice A. Contributeurs

Ophir Frieder (Illinois Institute of Technology) a collaboré au développement de l'algorithme de mémoire tampon.
Jeff Van Dyke a travaillé assez d'heures et écrit assez de texte pour être considéré comme auteur selon les vieilles règles.
Robert Fairlie-Cuninghame, Cullen Jennings, Jonathan Rosenberg, et nous, sommes les membres de l'équipe de conception de signalisation de stimulus d'application. Tous les membres de l'équipe ont contribué à ce travail. De plus, Jonathan Rosenberg a considéré DML comme établi dans son projet "Cadre pour une signalisation par stimulus dans SIP en utilisant le balisage".
Cette version de KPML a reçu une influence significative de MSCML [RFC4722], le langage de balisage de contrôle de serveur de support SnowShore. Jeff Van Dyke et Andy Spitzer ont été les principaux contributeurs de cet effort.
Rohan Mahy a fait une réorganisation significative du contenu, et a fourni un soutien moral considérable à la production de ce document.
Ceci dit, toutes les erreurs, mauvaises interprétations, ou fautes dans ce document incombent aux auteurs.

Appendice B. Remerciements

Hal Purdy et Eric Cheung des AT&T Laboratories ont apporté une aide immense par de nombreuses conversations et challenges.
Steve Fisher des AT&T Laboratories a suggéré la syntaxe de suppression de chiffre et effectué une excellente relecture du document.
Terence Lobo de SnowShore Networks a fait fonctionner le tout.
Jerry Kamitses, Swati Dhuleshia, Shaun Bharrat, Sunil Menon, et Bryan Hill ont aidé à préciser le comportement de la mémoire tampon et la syntaxe de DRegex.
Silvano Brewster et Bill Fenner des AT&T Laboratories et Joe Zebarth de Nortel ont apporté une aide considérable en précisant le texte et DRegex.

Bert Culpepper et Allison Mankin ont fait une révision sévère d'une première version de ce document.
Scott Hollenbeck a revu le XML et MIME. Tim Bray a soulevé le problème général de UTF-8 contre UTF-16 avec XML.

Adresse des auteurs

Eric Burger
Cantata Technology, Inc.
18 Keewaydin Dr.
Salem, NH 03079
USA
mél : eburger@cantata.com

Martin Dolly
AT&T Labs
mél : mdolly@att.com

Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est fourni par l'activité de soutien administatif (IASA) de l'IETF.