Groupe de travail Réseau
Request for Comments : 2046
RFC rendues obsolètes : 1521, 1522, 1590
Catégorie : Sur la voie de la normalisation

N. Freed, Innosoft
N. Borenstein, First Virtual
novembre 1996
Traduction Claude Brière de L'Isle

Extensions multi-objets de messagerie Internet (MIME) partie deux : types de supports

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.
(La présente traduction incorpore les errata existants 507, 508, , 509, 510, 511, 588, 589 et 4609.)

Résumé
Le STD 11, RFC 822, définit un protocole de représentation de message qui spécifie un nombre considérable de détails sur les en-têtes de message US-ASCII, et laisse le contenu du message, ou corps de message, comme du texte US-ASCII plat. Le présent ensemble de documents, appelé collectivement "extensions multi objets de messagerie Internet (MIME, *Multipurpose Internet Mail Extensions*) redéfinit le format des messages pour permettre :
(1) des corps de messages textuels dans des jeux de caractères autres que l'US-ASCII,
(2) un ensemble extensible de différents formats pour des corps de message non textuels,
(3) des corps de message multi parties, et
(4) des informations d'en-tête textuelles dans des jeux de caractères autres que l'US-ASCII.

Ces documents se fondent sur des travaux antérieurs documentés dans la RFC 934, STD 11, et dans la RFC 1049, mais les étendent et les révisent. Comme la RFC 822 en dit si peu sur les corps de message, ces documents sont largement orthogonaux (plutôt qu'une révision) à la RFC 822.

Le premier document de cet ensemble spécifie les divers en-têtes utilisés pour décrire la structure des messages MIME. Ce second document définit la structure générale du système de support MIME et définit un jeu initial de types de supports. Le troisième document, RFC 2047, décrit des extensions à la RFC 822 pour permettre des données de texte non US-ASCII dans les champs d'en-tête de messagerie Internet. Le quatrième document, RFC 2048, spécifie diverses procédures d'enregistrement par l'IANA pour les facilités relatives à MIME. Le cinquième et dernier document, RFC 2049, décrit les critères de conformité à MIMIE ainsi que des exemples illustrant les formats de message MIME, les remerciements et la bibliographie.

Ces documents sont des révisions des RFC 1521, et 1522, qui étaient elles-mêmes des révisions des RFC 1341 et 1342. Un appendice de la RFC 2049 décrit les différences et changements par rapport aux précédentes versions.

Table des matières

1. Introduction....2
2. Définition d'un type de support de niveau supérieur....2
3. Vue d'ensemble des types initiaux de support de niveau supérieur....2
4. Valeurs discrètes de types de supports....3
4.1 Type de support Text....3
4.2 Type de support Image....6
4.3 Type de support Audio....6
4.4 Type de support Video....6
4.5 Type de support Application....7
5. Valeurs de type de support composites....9
5.1 Type de support Multipart....9
5.2 Type de support Message....15
6. Valeurs expérimentales de type de support....22
7. Résumé....22

8. Considérations sur la sécurité.....23
9. Adresse des auteurs.....23
Appendice A -- Récapitulation de la grammaire.....23

1. Introduction

Le premier document de cet ensemble, la RFC 2045, définit un certain nombre de champs d'en-tête, incluant Content-Type. Le champ Content-Type est utilisé pour spécifier la nature des données dans le corps d'une entité MIME, en donnant les identifiants de type et de sous type de support, et en fournissant des informations auxiliaires qui peuvent être requises pour certains types de supports. Après les noms de type et sous type, le reste du champ d'en-tête est simplement un ensemble de paramètres, spécifiés dans une notation attribut/valeur. L'ordre des paramètres n'est pas significatif.

En général, le type de support de niveau supérieur est utilisé pour déclarer le type général des données, tandis que le sous type spécifie un format spécifique pour ce type de données. Donc, un type de support de "image/xyz" est suffisant pour dire à un agent d'utilisateur que les données sont une image, même si l'agent d'utilisateur n'a pas connaissance du format d'image "xyz" spécifique. De telles informations peuvent être utilisées, par exemple, pour décider si il faut ou non montrer à l'utilisateur les données brutes provenant d'un sous type non reconnu -- une telle action peut être raisonnable pour des sous types non reconnus de "text", mais pas pour des sous types non reconnus de "image" ou "audio". Pour cette raison, les sous types enregistrés de "text", "image", "audio", et "video" ne devraient pas contenir des informations incorporées qui sont réellement d'un type différent. De tels formats composés devrait être représentés en utilisant les types "multipart" ou "application".

Les paramètres sont des modificateurs du sous type de support, et à ce titre, n'affectent pas fondamentalement la nature du contenu. L'ensemble des paramètres significatifs dépend du type et sous type de support. La plupart des paramètres sont associés à un seul sous type spécifique. Cependant, un certain type de support de niveau supérieur peut définir des paramètres qui sont applicables à tous les sous types de ce type. Les paramètres peuvent être exigés par le type ou sous type de support qui les définit ou ils peuvent être facultatifs. Les mises en œuvre de MIME doivent aussi ignorer tous les paramètres dont elles ne reconnaissent pas le nom.

Le champ d'en-tête Content-Type de MIME et le mécanisme de type de support ont été conçus avec soin pour être extensibles, et il est prévu que l'ensemble des paires de type/sous type de support et leurs paramètres associés va croître de façon significative à l'avenir. Plusieurs autres facilités de MIME, comme les codages de transfert et les types d'accès "message/external-body" vont probablement avoir de nouvelles valeurs définies à l'avenir. Afin d'assurer que l'ensemble de ces valeurs est développé d'une manière ordonnée, bien spécifiée, et publique, MIME établit un processus d'enregistrement qui utilise l'autorité d'allocation des numéros de l'Internet (IANA) comme registre central pour les diverses zones d'extensibilité de MIME. Le processus d'enregistrement pour ces zones est décrit dans la RFC 2048.

Les sept types de support de niveau supérieur standard initiaux sont définis et décrits dans la suite du présent document.

2. Définition d'un type de support de niveau supérieur

Selon sa définition, un type de support de niveau supérieur consiste en :
(1) un nom et une description du type, incluant des critères pour qu'un type particulier se qualifie comme ce type,
(2) les noms et définitions de paramètres, si il en est, qui sont définis pour tous les sous types de ce type (incluant si ces paramètres sont exigés ou facultatifs),
(3) comment un agent d'utilisateur et/ou passerelle devrait traiter les sous types inconnus de ce type,
(4) des considérations générales sur les entités passerelles de ce type de niveau supérieur, s'il en est, et
(5) toutes restrictions sur les codages de transfert de contenu pour les entités de ce type de niveau supérieur.

3. Vue d'ensemble des types initiaux de support de niveau supérieur

Les cinq types de support de niveau supérieur distincts sont :

(1) text -- informations textuelles. Le sous type "plain" en particulier indique le texte pur qui ne contient aucune sorte de commande ou directive de formatage. Le texte pur est destiné à être affiché "tel quel". Aucun logiciel spécial n'est exigé pour avoir la pleine signification du texte, à part la prise en charge du jeu de caractères indiqué. D'autres sous types vont

être utilisés pour un texte enrichi dans des formes où le logiciel d'application peut améliorer l'apparence du texte, mais un tel logiciel ne doit pas être exigé pour avoir l'idée générale du contenu. Des sous types possibles de "text" incluent donc tout format de traitement de mot qui peut être lu sans renvoyer à un logiciel qui comprend le format. En particulier, les formats qui emploient des informations de formatage binaire incorporées ne sont pas considérés comme directement lisibles. Un sous type très simple et portable, "richtext", a été défini dans la RFC 1341, avec une révision ultérieure dans la RFC 1896 sous le nom de "enriched".

(2) image -- données d'image. "Image" exige un appareil d'affichage (comme un affichage graphique, une imprimante graphique, ou un télécopieur) pour voir les informations. Les sous types sont définis pour deux formats d'image largement utilisés, jpeg et gif.

(3) audio -- données audio. "Audio" exige un appareil de sortie audio (comme un haut-parleur ou un téléphone) pour "restituer" le contenu. Un sous type initial "basic" est défini dans ce document.

(4) video -- données vidéo. "Video" exige la capacité d'afficher des images animées, incluant normalement un matériel et un logiciel spécialisés. Un sous type initial "mpeg" est défini dans ce document.

(5) application -- certaines autres sortes de données, normalement des données ou informations binaires non interprétées à traiter par une application. Le sous type "octet-stream" est à utiliser dans le cas de données binaires non interprétées, et dans ce cas l'action recommandée la plus simple est d'offrir d'écrire les informations dans un fichier pour l'utilisateur. Le sous type "PostScript" est aussi défini pour le transport de matériel PostScript. D'autres utilisations prévues pour "application" incluent des feuilles de calcul, des données pour des systèmes de programmation fondés sur la messagerie, et des langages pour la messagerie "active" (de calcul) et les formats de traitement de texte qui ne sont pas directement lisibles. Noter que des considérations de sécurité peuvent exister pour certains types de données "application", notablement "application/PostScript" et toute forme de messagerie active. Ces questions sont discutées plus loin.

Les deux types de support composites de niveau supérieur sont :

(1) multipart -- données consistant en plusieurs entités de types de données indépendants. Quatre sous types sont initialement définis, incluant le sous type de base "mixed" qui spécifie un ensemble générique mixte de parties, "alternative" pour représenter les mêmes données dans de multiples formats, "parallel" pour les parties destinées à être vues simultanément, et "digest" pour les entités multi parties dans lesquelles chaque partie a un type par défaut de "message/rfc822".

(2) message -- un message encapsulé. Un corps de type de support "message" est lui-même tout ou partie d'un objet message d'une certaine sorte. De tels objets peuvent ou non à leur tour contenir d'autres entités. Le sous type "rfc822" est utilisé quand le contenu encapsulé est lui-même un message RFC 822. Le sous type "partial" est défini pour les messages partiels de la RFC 822, pour permettre la transmission fragmentée de corps qui sont estimés trop gros pour être passés en un seul morceau à travers les facilités de transport. Un autre sous type, "external-body", est défini pour spécifier les corps de grande taille par référence à une source de données externe.

On devrait noter que la liste des valeurs de type de support donnée ici pourra être augmentée à l'avenir, via les mécanismes décrits ci-dessus, et que l'ensemble des sous types est susceptible d'une croissance substantielle.

4. Valeurs distinctes de types de supports

Cinq des sept valeurs initiales de type de support se réfèrent à des corps distincts. Le contenu de ces types doit être traité par des mécanismes non MIME ; ils sont opaques aux processeurs MIME.

4.1 Type de support Text

Le type de support "text" est destiné à l'envoi de matériel principalement en forme textuelle. Un paramètre "charset" peut être utilisé pour indiquer le jeu de caractères du texte du corps pour les sous types "text", incluant notamment le sous type "text/plain", qui est un sous type générique pour le texte pur. "text/plain" ne fournit pas ni ne permet de commandes de formatage, de spécifications d'attributs de fonte, d'instructions de traitement, de directives d'interprétation, ni de balisage de contenu. "text/plain" est vu simplement comme une séquence linéaire de caractères, éventuellement interrompue par des coupures de ligne ou de page. "text/plain" peut permettre la mise en pile de plusieurs caractères dans la même position dans le texte. Le texte pur dans des scripts comme l'arabe et l'hébreu peut aussi inclure des facilités qui permettent le mélange arbitraire de segments de texte avec des directions d'écriture opposées.

Au delà du texte pur, il y a de nombreux formats pour représenter ce qui peut être appelé du "texte enrichi". Une caractéristique intéressante de nombre de ces représentations est qu'elles sont dans une certaine mesure lisibles même sans le logiciel qui les interprète. Il est donc utile de les distinguer, au plus haut niveau, des données non lisibles comme les images, l'audio, ou le texte représenté sous une forme non lisible. En l'absence du logiciel d'interprétation approprié, il est raisonnable de montrer les sous types de "text" à l'utilisateur, alors qu'il n'est pas raisonnable de le faire avec la plupart des données non textuelles. De telles données textuelles formatées devraient être représentées en utilisant des sous types de "text".

4.1.1 Représentation des coupures de ligne

La forme canonique de tout sous type MIME "text" DOIT toujours représenter une coupure de ligne par une séquence CRLF. De même, toute occurrence de CRLF dans un "text" MIME DOIT représenter une coupure de ligne. L'utilisation de CR et LF en dehors des séquences de coupure de ligne est aussi interdite.

Cette règle s'applique sans considération du format ou jeu de caractères impliqué.

Note : l'interprétation appropriée des coupures de ligne quand un corps est affiché dépend du type de support. En particulier, alors qu'il est approprié de traiter une coupure de ligne comme une transition vers une nouvelle ligne lors de l'affichage d'un corps "text/plain", ce traitement est en fait incorrect pour les autres sous types de "text" comme "text/enriched" [RFC1896]. De même, la question de savoir si les coupures de ligne devraient ou non être ajoutées durant les opérations d'affichage est aussi une fonction du type de support. Il ne devrait pas être nécessaire d'ajouter de coupure de ligne pour afficher correctement "text/plain", tandis que l'affichage approprié de "text/enriched" exige l'ajout approprié des coupures de ligne.

Note : certains protocoles définissent une longueur maximum de ligne. Par exemple. SMTP [RFC0821] permet un maximum de 998 octets avant la prochaine séquence CRLF. Pour être transportées par de tels protocoles, les données qui comportent de trop longs segments sans séquence CRLF doivent être codées avec un codage de transfert de contenu convenable.

4.1.2 Paramètre Charset

Un paramètre critique qui peut être spécifié dans le champ Content-Type pour des données "text/plain" est le jeu de caractères. Ceci est spécifié avec un paramètre "charset", comme dans :

```
Content-type: text/plain; charset=iso-8859-1
```

À la différence de certaines autres valeurs de paramètres, les valeurs du paramètre charset NE SONT PAS sensibles à la casse. Le jeu de caractères par défaut, qui doit être supposé en l'absence d'un paramètre charset, est US-ASCII.

La spécification de tous futurs sous types de "text" doit spécifier si ils doivent ou non utiliser aussi un paramètre "charset", et peut éventuellement restreindre aussi ses valeurs. Pour les autres sous types de "text" que "text/plain", la sémantique du paramètre "charset" devrait être définie comme étant identique à celle spécifiée ici pour "text/plain", c'est-à-dire, le corps consiste entièrement en caractères du jeu de caractères en question. En particulier, la définition des futurs sous types "text" devrait faire très attention aux implications des jeux de caractères multi octets pour leur définition de sous type.

Le paramètre charset pour les sous types de "text" donne un nom de jeu de caractères, comme "jeu de caractères" est défini dans la RFC2045. Les règles concernant les coupures de ligne détaillées dans la section précédente doivent aussi être observées -- un jeu de caractères dont la définition ne se conforme pas à ces règles ne peut pas être utilisé dans un sous type MIME "text".

Une liste initiale de noms de jeux de caractères prédéfinis se trouve à la fin de cette section. Des jeux de caractères supplémentaires peuvent être enregistrés auprès de l'IANA.

D'autres types de support que les sous types de "text" pourraient choisir d'employer le paramètre "charset" comme défini ici, mais en supprimant la restriction de CRLF/coupure de ligne. Donc, tous les jeux de caractères qui se conforment à la définition générale de "jeu de caractères" dans la RFC 2045 peuvent être enregistrés pour l'usage de MIME.

Noter que si le jeu de caractères spécifié inclut des caractères 8-bit et que de tels caractères sont utilisés dans le corps, un champ d'en-tête Content-Transfer-Encoding et un codage correspondant sur les données sont exigés afin de transmettre le

corps via des protocoles de transfert de messagerie, comme SMTP [RFC0821].

Le jeu de caractères par défaut, US-ASCII, a été l'objet dans le passé de certaines confusions et ambiguïtés. Non seulement il y avait des ambiguïtés dans la définition, mais il y a eu de larges variations dans la pratique. Afin d'éliminer ces ambiguïtés et variations à l'avenir, il est fortement recommandé que les nouveaux agents d'utilisateur spécifient explicitement un jeu de caractères comme paramètre de type de support dans le champ d'en-tête Content-Type. "US-ASCII" n'indique pas un jeu de caractères arbitraire de 7-bit, mais spécifie que tous les octets dans le corps doivent être interprétés comme des caractères conformes au jeu de caractères US-ASCII. Les versions nationales et orientées application de ISO 646 [ISO-646] NE sont généralement PAS identiques à l'US-ASCII, et dans ce cas, leur utilisation dans la messagerie Internet est explicitement déconseillée. L'omission du jeu de caractères ISO 646 du présent document est délibérée à cet égard. Le nom de jeu de caractères de "US-ASCII" se réfère explicitement au jeu de caractères défini dans la norme ANSI X3.4-1986 [US-ASCII]. La nouvelle version de référence internationale (IRV) de l'édition 1991 de ISO 646 est identique à l'US-ASCII. Le nom de jeu de caractères "ASCII" est réservé et ne doit en aucun cas être utilisé.

Note : la RFC 821 spécifie explicitement "ASCII", et fait référence à une version antérieure de la norme américaine. Dans la mesure où la raison de la spécification d'un type de support et d'un jeu de caractères est de permettre au receveur de déterminer sans ambiguïté comme l'envoyeur entendait que soit interprété le message codé, supposer autre chose que du "strict ASCII" par défaut ferait courir le risque de changements involontaires et incompatibles à la sémantique des messages transmis. Cela implique aussi que les messages qui contiennent des caractères codés selon d'autres versions de ISO 646 que US-ASCII et l'IRV de 1991, ou utilisent des procédures de changement de code (par exemple, celles de ISO 2022) ainsi que des codages de caractères sur 8bit ou plusieurs octets DOIVENT utiliser une spécification de jeu de caractères appropriée pour être cohérents avec MIME.

Le jeu de caractères US-ASCII complet est décrit dans la norme ANSI X3.4-1986. Noter que les caractères de contrôle incluant DEL (0-31, 127) n'ont pas de signification définie à part la combinaison CRLF (valeurs US-ASCII 13 et 10) pour indiquer une nouvelle ligne. Deux des caractères ont de fait une signification largement utilisée : FF (12) signifie souvent "commencer le texte suivant sur une nouvelle page"; et TAB ou HT (9) signifie souvent (mais pas toujours) "déplacer le curseur à la prochaine colonne disponible après la position actuelle lorsque le numéro de colonne est un multiple de 8 (en comptant la première colonne comme colonne 0)". À part ces conventions, toute utilisation de caractères de contrôle ou DEL dans un corps doit se produire :
(1) parce qu'un sous type de texte autre que "plain" donne spécifiquement une signification supplémentaire, ou
(2) dans le contexte d'un accord privé entre l'envoyeur et le receveur. De tels accords privés sont déconseillés et devraient être remplacés par les autres capacités de ce document.

Note : une énorme prolifération de jeux de caractères existe au delà de l'US-ASCII. Un grand nombre de jeux de caractères qui se chevauchent partiellement ou totalement N'est PAS une bonne chose. Un SEUL jeu de caractères qui pourrait être universellement utilisé pour représenter toutes les langues du monde dans la messagerie Internet serait préférable. Malheureusement, la pratique existante dans les diverses communautés semble indiquer la continuation de l'utilisation de multiples jeux de caractères pour l'avenir proche. Un petit nombre de jeux de caractères standard est donc défini pour l'usage dans l'Internet dans le présent document.

Les valeurs définies de jeux de caractères sont :
(1) US-ASCII -- comme défini dans la norme ANSI X3.4-1986 [US-ASCII].
(2) ISO-8859-X -- où "X" est à remplacer, comme nécessaire, par les parties de la norme ISO-8859 [ISO-8859]. Noter que les jeux de caractères ISO 646 ont été délibérément omis en faveur de leurs remplacements de ISO 8859, qui sont les jeux de caractères conçus pour la messagerie Internet. À la publication du présent document, les valeurs légitimes pour "X" sont les chiffres de 1 à 10.

Les caractères dans la gamme 128 à 159 n'ont pas de signification allouée dans la norme ISO-8859-X. Les caractères qui ont des valeurs en dessous de 128 dans ISO-8859-X ont la même signification que dans l'US-ASCII.

La partie 6 de ISO 8859 (alphabet latin/arabe) et la partie 8 (alphabet latin/hébreu) incluent à la fois des caractères pour lesquels la direction normale d’écriture est de droite à gauche et des caractères pour lesquels c'est de gauche à droite, mais elles ne définissent pas de méthode d'ordre canonique pour représenter du texte bidirectionnel. Les valeurs de jeu de caractère "ISO-8859-6" et "ISO-8859-8" spécifient cependant que la méthode visuelle est utilisée [RFC1556].

Tous ces jeux de caractère sont utilisés comme purs jeux 7bit ou 8bit sans aucune fonction de décalage ou d'échappement. La signification des séquences de décalage et d'échappement dans ces jeux de caractères n'est pas définie.

Les jeux de caractères spécifiés ci-dessus sont ceux qui étaient relativement incontestables durant la rédaction de MIME. Le présent document n'entérine l'utilisation d'aucun jeu de caractères particulier autre que l'US-ASCII, et reconnaît que la future évolution des jeux de caractères mondiaux reste peu claire.

Noter que le jeu de caractères utilisé, si c'est autre chose que l'US-ASCII, doit toujours être explicitement spécifié dans le champ Content-Type.

Aucun nom de jeu de caractères autre que ceux définis ci-dessus ne peut être utilisé dans la messagerie Internet sans la publication d'une spécification formelle et son enregistrement par l'IANA, ou par accord privé, et dans ce cas, le nom du jeu de caractères doit commencer par "X-".

Il est déconseillé de définir de nouveaux jeux de caractères sans absolue nécessité.

Le paramètre "charset" a été principalement défini pour les besoins des données de texte, et est décrit dans cette section pour cette raison. Cependant, il est concevable que des données non de texte puissent aussi souhaiter spécifier une valeur de "charset" pour un certain objet, et dans ce cas la même syntaxe et les mêmes valeurs devraient être utilisées.

En général, le logiciel de composition devrait toujours utiliser le "plus petit dénominateur commun" possible de jeu de caractères. Par exemple, si un corps contient seulement des caractères US-ASCII, il DEVRAIT être marqué comme étant dans le jeu de caractères US-ASCII, et non ISO-8859-1, qui, comme toute la famille des jeux de caractères ISO-8859, est un sur-ensemble de US-ASCII. Plus généralement, si un jeu de caractères largement utilisé est un sous-ensemble d'un autre jeu de caractères, et si un corps contient seulement des caractères dans le sous-ensemble largement utilisé, il devrait être étiqueté comme étant dans ce sous-ensemble. Cela va augmenter les chances que le receveur soit capable de voir correctement l'entité résultante.

4.1.3 Sous type Plain

Le plus simple et plus important sous type de "text" est "plain". Cela indique du texte pur qui ne contient aucune commande ou directive de formatage. Le texte pur est destiné à être affiché "tel quel", c'est-à-dire, aucune interprétation de commandes de formatage incorporées, de spécifications d'attributs de fonte, d'instructions de traitement, de directives d’interprétation, ou de balisage de contenu ne devraient être nécessaires pour un affichage approprié. Le type de support de "text/plain; charset=us-ascii" par défaut pour la messagerie Internet décrit la pratique existante de l'Internet. C'est-à-dire que c'est le type de corps défini par la RFC 822.

Aucun autre sous type "text" n'est défini par le présent document.

4.1.4 Sous types non reconnus

Les sous types non reconnus de "text" devraient être traités comme du sous type "plain" pour autant que la mise en œuvre MIME sache comment traiter le jeu de caractères. Les sous types non reconnus qui spécifient aussi un jeu de caractères non reconnu devraient être traités comme "application/octet-stream".

4.2 Type de support Image

Un type de support de "image" indique que le corps contient une image. Le sous type désigne le format d'image spécifique. Ces noms ne sont pas sensibles à la casse. Un sous type initial est "jpeg" pour le format JPEG qui utilise le codage JFIF [JPEG].

La liste des sous types "image" qu'on donne ici n'est ni exclusive ni exhaustive, et elle est supposée croître avec des types qui seront enregistrés par l'IANA, comme décrit dans la RFC 2048.

Les sous types non reconnus de "image" devraient au minimum être traités comme "application/octet-stream". Les mises en œuvre peuvent choisir de passer des sous types de "image" qu'ils ne reconnaissent pas précisément à une application sûre et robuste de visionnage d'image non spécialisée, si une telle application est disponible.

Note : Utiliser de cette façon une application de visionnage d'image non spécialisée hérite des problèmes de sécurité du type le plus dangereux pris en charge par l'application.

4.3 Type de support Audio

Un type de support de "audio" indique que le corps contient des données audio. Bien qu'il n'y ait pas encore de consensus sur un format audio "idéal" à utiliser sur les ordinateurs, il y a un pressant besoin d'un format capable de fournir un

comportement interopérable.

Le sous type initial de "basic" est spécifié pour satisfaire aux exigences en fournissant un plus petit dénominateur commun de format audio absolument minimal. On s'attend à ce que des formats plus riches pour un audio de qualité supérieure et/ou de moindre bande passante soient définis dans des documents ultérieurs.

Le contenu du sous type "audio/basic" est un seul canal audio codé en utilisant le RNIS 8bit en loi μ [PCM] au taux d'échantillonnage de 8 000 Hz.

Les sous types non reconnus de "audio" devraient au minimum être traités comme "application/octet-stream". Les mises en œuvre peuvent choisir de passer des sous types de "audio" qu'elles ne reconnaissent pas spécifiquement à une application d'exécution audio robuste non spécialisée, si une telle application est disponible.

4.4 Type de support Video

Un type de support de "video" indique que le corps contient une image qui varie dans le temps, éventuellement avec de la couleur et du son coordonné. Le terme "video" est utilisé dans son sens le plus générique, plutôt que par référence à une technologie ou un format particuliers, et n'est pas destiné à empêcher des sous types comme des dessins animés à codage compact. Le sous type "mpeg" se réfère à la vidéo codée conformément à la norme MPEG [MPEG].

Noter que bien qu'en général le présent document déconseille fortement le mélange de plusieurs supports dans un seul corps, il est reconnu que de nombreux formats soit-disant vidéo incluent une représentation pour de l'audio synchronisé, et ceci est explicitement permis pour les sous types de "video".

Les sous types non reconnus de "video" devraient au minimum être traités comme un "application/octet-stream". Les mises en œuvre peuvent choisir de passer des sous types de "video" qu'elles ne reconnaissent pas spécifiquement à une application robuste d'affichage générique de vidéo, si une telle application est disponible.

4.5 Type de support Application

Le type de support "application" est à utiliser pour des données discrètes qui ne tiennent dans aucune autre catégorie, et particulièrement pour des données à traiter par un certain type de programme d'application. Ce sont des informations qui doivent être traitées par une application avant qu'elles soient visibles ou utilisables par un utilisateur. Les utilisations attendues pour le type de support "application" incluent des transferts de fichiers, des feuilles de calcul, des données pour les systèmes de programmation fondés sur la messagerie, et des langages pour du matériel "actif" (du point de vue du calcul). (Ces dernières, en particulier, peuvent poser des problèmes de sécurité qui doivent être compris par les mises en œuvre, et sont considérés en détail dans la discussion du type de support "application/PostScript".)

Par exemple, un programmeur de réunion pourrait définir une représentation standard pour des informations sur les dates de réunion proposées. Un agent d'utilisateur intelligent va utiliser ces informations pour conduire un dialogue avec l'utilisateur, et pourrait alors envoyer du matériel supplémentaire sur la base de ce dialogue. Plus généralement, il y a eu plusieurs langages de messagerie "active" qui ont été développés dans lesquels les programmes dans un langage convenablement spécialisé sont transportés à une localisation distante et fonctionnent automatiquement dans l'environnement de l'utilisateur.

De telles applications peuvent être définies comme des sous types du type de support "application". Le présent document définit deux sous types : octet-stream, et PostScript.

Le sous type de "application" va souvent être soit le nom, soit inclure des parties du nom de l'application pour laquelle les données sont destinées. Cela ne signifie cependant pas que tout nom de programme d'application peut être utilisé librement comme sous type de "application".

4.5.1 Sous type Octet-Stream

Le sous type "octet-stream" est utilisé pour indiquer qu'un corps contient des données binaires arbitraires. L'ensemble des paramètres actuellement définis est :

(1) TYPE -- le type général ou catégorie des données binaires. Ceci est destiné à des informations pour le receveur humain plutôt que pour un traitement automatique.

(2) PADDING -- le nombre de bits de bourrage qui ont été ajoutés au flux binaire constituant le contenu réel pour produire les données en 8bit en mode octet encloses. Ceci est utile pour inclure un flux de bits dans un corps quand le nombre total de bits n'est pas un multiple de 8.

Ces deux paramètres sont facultatifs.

Un paramètre supplémentaire, "CONVERSIONS", a été défini dans la RFC 1341 mais a été supprimé depuis. La RFC1341 définissait aussi l'utilisation d'un paramètre "NAME" qui donnait un nom de fichier dont l'utilisation était suggérée si les données étaient à écrire dans un fichier. Cela a été déconseillé en anticipation d'un champ d'en-tête Content-Disposition séparé, qui devait être défini dans une RFC à venir.

L'action recommandée pour une mise en œuvre qui reçoit une entité "application/octet-stream" est simplement d'offrir de mettre les données dans un fichier, en défaisant tout Content-Transfer-Encoding, ou peut-être de l'utiliser en entrée d'un processus spécifié par l'utilisateur.

Pour réduire le danger de transmettre des programmes malveillants, il est fortement recommandé que les mises en œuvre NE METTENT PAS en œuvre un mécanisme de recherche de chemin par lequel un programme arbitraire nommé dans le paramètre Content-Type (par exemple, un paramètre "interpreter=") est trouvé et exécuté en utilisant le corps du message comme entrée.

4.5.2 Sous type PostScript

Un type de support de "application/postscript" indique un programme PostScript. Actuellement, deux variantes du langage PostScript sont permis ; la variante originale de niveau 1 est décrite dans [POSTSCRIPT] et la variante plus récente de niveau 2 est décrite dans [POSTSCRIPT2].

PostScript est une marque déposée de Adobe Systems, Inc. L'utilisation du type de support MIME "application/postscript" implique la reconnaissance de cette marque commerciale et de tous les droits qu'elle comporte.

La définition du langage PostScript donne des facilités pour l'étiquetage interne des caractéristiques spécifiques du langage qu'utilise un certain programme. Cet étiquetage, appelé "les conventions de structure de document PostScript", ou DSC, est très général et donne des informations plus substantielles que juste le niveau de langage. L'utilisation des conventions de structuration de document, bien que non exigées, sont fortement recommandées comme aide à l'interopérabilité. Les documents qui n'ont pas les conventions de structuration appropriées ne peuvent pas être vérifiés pour voir si ils vont ou non fonctionner dans un certain environnement. À ce titre, certains systèmes peuvent supposer le pire et refuser de traiter les documents non structurés.

L'exécution d'interpréteurs PostScript génériques entraîne des risques sérieux pour la sécurité, et il est déconseillé aux mises en œuvre d'envoyer simplement des corps PostScript à des interpréteurs inconnus. Bien qu'il soit généralement sûr d'envoyer du PostScript à une imprimante, car le potentiel de dommages est largement contraint par les environnement normaux d'imprimante, les mises en œuvre devraient considérer tout ce qui suit avant d'ajouter un affichage interactif de corps PostScript à leurs lecteurs MIME.

Le reste de cette section souligne certains, mais probablement pas tous, des problèmes possibles avec le transport des entités PostScript.

(1) Des opérations dangereuses dans le langage PostScript incluent, mais peuvent ne pas y être limitées, les opérateurs PostScript "deletefile", "renamefile", "filenameforall", et "file". "File" n'est dangereux que lorsque appliqué à quelque chose d'autre que des entrées ou sorties standard. Les mises en œuvre peuvent aussi définir des opérateurs de fichier non standard supplémentaires ; ceux-ci peuvent aussi faire peser une menace. "Filenameforall", l'opérateur de recherche générique de fichier, peut apparaître à première vue comme inoffensif. Noter, cependant, que cet opérateur a le potentiel de révéler des informations sur les fichiers auxquels le receveur a accès, et cette information peut elle-même être sensible. Les envoyeurs de messages devraient éviter d'utiliser les opérateurs de fichier qui sont potentiellement dangereux, car ces opérateurs vont probablement être indisponibles dans une mise en œuvre sécurisée de PostScript. Le logiciel de réception et d'affichage de messages devrait soit désactiver complètement tous les opérateurs de fichiers potentiellement dangereux, soit porter une attention particulière à ne pas déléguer d'autorité particulière à leur fonctionnement. Ces opérateurs devraient être vus comme étant effectués par une agence extérieure quand ils interprètent des documents PostScript. Une telle désactivation et/ou vérification devrait être faire complètement en dehors de la portée du langage PostScript lui-même ; on devrait s'assurer qu'aucune méthode n'existe pour réactiver les versions de pleine fonction de ces opérateurs.

(2) Le langage PostScript fournit des facilités pour sortir de la boucle de l'interpréteur ou serveur normal. Les changements faits dans cet environnement "externe" sont conservées généralement d'un document à l'autre, et peuvent dans certains cas être conservés ad vitam æternam dans une mémoire non volatile. Les opérateurs associés à la sortie de la boucle

d'interpréteur peuvent potentiellement interférer avec le traitement du document suivant. À ce titre, leur utilisation non contrôlée constitue une menace de déni de service. Les opérateurs PostScript qui sortent de la boucle d'interpréteur incluent, mais peuvent ne pas y être limités, les opérateurs "exitserver" et "startjob". Le logiciel d'envoi de messages ne devrait pas générer de PostScript qui dépende de la sortie de la boucle d'interpréteur pour fonctionner, car la capacité de sortie ne va probablement pas être disponible dans une mise en œuvre PostScript sécurisée. Le logiciel de réception et d'affichage de messages devrait désactiver complètement la capacité de faire des changements à l'environnement PostScript en éliminant ou désactivant les opérateurs "startjob" et "exitserver". Si ces opérations ne peuvent pas être éliminées ou complètement désactivées, le mot de passe qui leur est associé devrait au moins être réglé à une valeur difficile à deviner.

(3) PostScript fournit des opérateurs pour régler les paramètres à l'échelle du système et spécifiques de l'appareil. Ces réglages de paramètres peut être conservés d'une tâche à l'autre et peuvent faire peser une menace potentielle pour le fonctionnement correct de l'interpréteur. Les opérateurs PostScript qui établissent les paramètres du système et de l'appareil incluent, mais peuvent ne pas y être limités, les opérateurs "setsystemparams" et "setdevparams". Le logiciel d'envoi de message ne devrait pas générer de PostScript qui dépende de l'établissement de paramètres du système ou de l'appareil pour fonctionner correctement. La capacité d'établir ces paramètres sera probablement indisponible dans une mise en œuvre sécurisée de PostScript. Le logiciel de réception et d'affichage de message devrait désactiver la capacité de changer les paramètres du système et de l'appareil. Si ces opérateurs ne peuvent pas être complètement désactivés, le mot de passe qui leur est associé devrait au moins être réglé à une valeur difficile à deviner.

(4) Certaines mises en œuvre PostScript fournissent des facilités non standard pour le chargement direct et l'exécution du code machine. De telles facilités sont assez évidemment ouvertes à des abus substantiels. Le logiciel d'envoi de message ne devrait pas utiliser de telles caractéristiques. En plus d'être totalement spécifiques du matériel, elles vont aussi être probablement indisponibles dans les mises en œuvre sécurisées de PostScript. Le logiciel de réception et d'affichage de message ne devrait pas permettre que de tels opérateurs soient utilisés si ils existent.

(5) PostScript est un langage extensible, et la plupart, sinon toutes ses mises en œuvre fournissent un certain nombre de leurs propres extensions. Le présent document ne traite pas explicitement de telles extensions car elles constituent un facteur inconnu. Le logiciel d'envoi de message ne devrait pas utiliser d'extensions non standard ; elles vont probablement être absentes de certaines mises en œuvre. Le logiciel de réception et d'affichage de message devrait s'assurer que tous les opérateurs PostScript non standard sont sûrs et ne présentent aucune sorte de menace.

(6) Il est possible d'écrire du PostScript qui consomme des quantités énormes de diverses ressources système. Il est aussi possible d'écrire des programmes PostScript qui sont indéfiniment en boucle. Ces deux types de programmes ont la capacité de causer des dommages si ils sont envoyés à des receveurs qui ne se méfient pas. Le logiciel d'envoi de messages devrait éviter de construire et disséminer de tels programmes, qui sont antisociaux. Le logiciel de réception et d'affichage de messages devrait fournir des mécanismes appropriés pour interrompre le traitement après qu'un délai raisonnable s'est écoulé. De plus, les interpréteurs PostScript devraient être limités à la consommation de seulement une quantité raisonnable de ressources du système.

(7) Il est possible d'inclure des informations binaires brutes dans PostScript sous des formes variées. Ceci n'est pas recommandé pour l'utilisation de la messagerie Internet, à la fois parce que ce n'est pas pris en charge par tous les interpréteurs PostScript et parce que cela complique significativement l'utilisation d'un codage de transfert de contenu MIME. (Sans de telles données binaires, PostScript peut normalement être vu comme des données en mode ligne. Le traitement des séquences CRLF devient extrêmement problématique si des données binaires et en mode ligne sont mélangées dans un seul flux de données Postscript.)

(8) Finalement, des erreurs peuvent exister dans certains interpréteurs PostScript qui pourraient éventuellement être exploitées pour obtenir un accès non autorisé au système d'un receveur. À part de noter cette possibilité, il n'y a pas d'action spécifique à prendre pour empêcher cela, à part de corriger ces erreurs à temps quand on les trouve.

4.5.3 Autres sous types Application

On s'attend à ce que de nombreux autres sous types de "application" soient définis à l'avenir. Les mises en œuvre MIME doivent au minimum traiter tous les sous types non reconnus comme étant équivalents à "application/octet-stream".

5. Valeurs de type de support composites

Les deux valeurs restantes des sept valeurs initiales de Content-Type se réfèrent à des entités composites. Les entités

composites sont traitées en utilisant les mécanismes MIME -- un processeur MIME traite normalement le corps directement.

5.1 Type de support Multipart

Dans le cas d'entités multi parties, dans lesquelles un ou plusieurs ensembles différents de données sont combinés en un seul corps, un champ de type de support "multipart" doit apparaître dans l'en-tête de l'entité. Le corps doit alors contenir une ou plusieurs parties du corps, chacune précédée par une ligne de délimiteur de frontière, et la dernière suivie par une ligne de délimiteur de frontière de clôture. Après sa ligne de délimiteur de frontière, chaque partie de corps consiste alors en une zone d'en-tête, une ligne blanche, et une zone de corps. Donc une partie de corps a une syntaxe similaire à celle d'un message RFC 822, mais a une signification différente.

Une partie de corps est une entité et donc N'EST PAS à interpréter comme étant en fait un message RFC 822. Pour commencer, AUCUN champ d'en-tête n'est en fait exigé dans les parties de corps. Une partie de corps qui commence par une ligne blanche est donc permise et est une partie de corps pour laquelle toutes les valeurs par défaut doivent être supposées. Dans ce cas, l'absence d'un en-tête Content-Type indique généralement que le corps correspondant a un content-type de "text/plain; charset=US-ASCII".

Les seuls champs d'en-tête qui ont une signification définie pour les parties de corps sont ceux dont les noms commencent par "Content-". Tous les autres champs d'en-tête peuvent être ignorés dans les parties de corps. Bien qu'ils devraient généralement être conservés si possible, ils peuvent être éliminés par les passerelles si nécessaire. Il est permis à d'autres champs d'apparaître dans les parties de corps mais ils ne doivent pas en être dépendants. Des champs "X-" peuvent être créés pour des besoins expérimentaux ou privés, tout en sachant que les informations qu'ils contiennent peuvent être perdues à une certaine passerelle.

Note : La distinction entre un message RFC 822 et une partie de corps est subtile, mais importante. Une passerelle entre l'Internet et une messagerie X.400, par exemple, doit être capable de faire la différence entre une partie de corps qui contient une image et une partie de corps qui contient un message encapsulé, dont le corps est une image JPEG. Afin de représenter cette dernière, la partie de corps doit avoir "Content-Type: message/rfc822", et son corps (après la ligne blanche) doit être le message encapsulé, avec son propre champ d'en-tête "Content-Type: image/jpeg". L'utilisation d'une syntaxe similaire facilite la conversion des messages en parties de corps, et vice versa, mais la distinction entre les deux doit être comprise par les mises en œuvre. (Pour le cas particulier dans lequel les parties sont en fait des messages, un sous type "digest" est aussi défini.)

Comme on l'a dit précédemment, chaque partie de corps est précédée par une ligne de délimiteur de frontière qui contient le délimiteur de frontière. Le délimiteur de frontière NE DOIT PAS apparaître dans une des parties encapsulées, sur une ligne par elle-même ou comme préfixe d'une ligne. Cela implique qu'il est crucial que l'agent composeur soit capable de choisir et spécifier une valeur unique de paramètre de frontière qui ne contienne pas comme préfixe la valeur d'un paramètre frontière d'une multi partie enclosante.

Tous les sous types présents et futurs du type "multipart" doivent utiliser une syntaxe identique. Des sous types peuvent différer par leur sémantique, et peuvent imposer des restrictions supplémentaires à la syntaxe, mais doivent se conformer à la syntaxe requise pour le type "multipart". Cette exigence assure que tous les agents d'utilisateur conformes vont au moins être capables de reconnaître et séparer les parties de toute entité multi parties, même celles d'un sous type non reconnu.

Comme indiqué dans la définition du champ Content-Transfer-Encoding [RFC2045], aucun codage autre que "7bit", "8bit", ou "binary" n'est permis pour les entités du type "multipart". Les délimiteurs de frontière de "multipart" et les champs d'en-tête sont toujours représentés comme de l'US-ASCII 7bit dans tous les cas (bien que les champs d'en-tête puissent coder un texte d'en-tête non US-ASCII selon la RFC 2047) et les données au sein des parties du corps peuvent être codées partie par partie, avec des champs Content-Transfer-Encoding pour chaque partie de corps appropriée.

5.1.1 Syntaxe commune

Ce paragraphe définit une syntaxe commune pour les sous types de "multipart". Tous les sous types de "multipart" doivent utiliser cette syntaxe. Un simple exemple d'un message multi parties est aussi donné dans ce paragraphe. Un exemple d'un message multi parties plus complexe est donné dans la RFC 2049.

Le champ Content-Type pour les entités multi parties exige un paramètre, "boundary". La ligne de délimiteur de frontière est alors définie comme une ligne consistant uniquement en deux caractères tiret ("-", valeur décimale 45) suivis par la valeur du paramètre de frontière à partir du champ d'en-tête Content-Type, une espace blanche linéaire facultative et un

CRLF de terminaison.

Note : les tirets sont pour la compatibilité avec la méthode antérieure de la RFC 934 pour l'encapsulation de message, et pour faciliter la recherche des frontières dans certaines mises en œuvre. Cependant, on devrait noter que les messages multipart NE SONT PAS complètement compatibles avec les encapsulations de la RFC 934 ; en particulier, ils n'obéissent pas aux conventions de guillemets de la RFC 934 pour les lignes incorporées qui commencent par des tirets. Ce mécanisme a été choisi plutôt que le mécanisme de la RFC 934 parce que ce dernier cause la croissance des lignes à chaque niveau de guillemets. La combinaison de cette croissance avec le fait que les mises en œuvre de SMTP coupent les longues lignes rend le mécanisme de la RFC 934 peu convenable pour l'utilisation dans le cas où une structuration de multi parties très incorporées serait désirée.

Avertissement pour les mises en œuvre : la grammaire des paramètres sur le champ Content-type est telle qu'il est souvent nécessaire d'enclore les valeurs de paramètre de frontière entre des guillemets sur la ligne de Content-type. Ceci n'est pas toujours nécessaire, mais ne cause jamais de problèmes. Les mises en œuvre devraient être sûres d'étudier attentivement la grammaire afin d'éviter de produire des champs Content-type invalides. Donc, un champ d'en-tête Content-Type "multipart" normal devrait ressembler à ceci :

Content-Type: multipart/mixed; boundary=gc0p4Jq0M2Yt08j34c0p

Mais le suivant n'est pas valide :

Content-Type: multipart/mixed; boundary=gc0pJq0M:08jU534c0p

(à cause des deux points) et doit à la place être représenté comme

Content-Type: multipart/mixed; boundary="gc0pJq0M:08jU534c0p"

Cette valeur de Content-Type indique que le contenu consiste en une ou plusieurs parties, chacune avec une structure syntaxiquement identique à celle du message RFC 822, sauf que la zone d'en-tête peut être complètement vide, et que les parties sont chacune précédée de la ligne --gc0pJq0M:08jU534c0p

Le délimiteur de frontière DOIT se produire au début d'une ligne, c'est-à-dire, à la suite d'un CRLF, et le CRLF initial est considéré être attaché à la ligne de délimiteur de frontière plutôt que faisant partie de la partie précédente. La frontière peut être suivie par zéro, un ou plusieurs caractères d'espace blanche linéaire. Elle est alors terminée soit par un autre CRLF et les champs d'en-tête pour la prochaine partie, soit par deux CRLF, et dans ce cas il n'y a pas de champs d'en-tête pour la prochaine partie. Si aucun champ Content-Type n'est présent, il est supposé être "message/rfc822" dans un "multipart/digest" et "text/plain" autrement.

Note : le CRLF qui précède la ligne de délimiteur de frontière est conceptuellement attachée à la frontière de sorte qu'il est possible d'avoir une partie qui ne se termine pas par un CRLF (coupure de ligne). Les parties de corps qui doivent être considérées se terminer par une coupure de ligne doivent donc avoir deux CRLF précédant la ligne de délimiteur de frontière, dont le premier fait partie de la partie de corps précédente, et le second de la frontière d'encapsulation.

Les délimiteurs de frontière ne doivent pas apparaître au sein du matériel encapsulé, et ne doivent pas être de plus de 70 caractères, sans compter les deux tirets en tête.

La ligne de délimiteur de frontière suivant la dernière partie de corps est un délimiteur distinctif qui indique qu'aucune autre partie de corps ne va suivre. Une telle ligne de délimiteur est identique aux lignes de délimiteur précédentes, avec l'ajout de deux tirets de plus après la valeur de paramètre de frontière : --gc0pJq0M:08jU534c0p--

Note de mise en œuvre : Les comparaisons de chaînes de frontière doivent comparer la valeur de la frontière avec le commencement de chaque ligne candidate. Une correspondance exacte de la ligne candidate entière n'est pas exigée ; il est suffisant que la frontière apparaisse entièrement à la suite du CRLF.

Il apparaît qu'il y a de la place pour des informations supplémentaires avant la première ligne de délimiteur de frontière et à la suite de la ligne finale de délimiteur de frontière. Ces zones devraient généralement être laissées blanches, et les mises en œuvre doivent ignorer tout ce qui apparaît avant la première ligne de délimiteur de frontière ou après la dernière.

Note : ces zones de "préambule" et "épilogue" ne sont généralement pas utilisées à cause de l'absence de conventions appropriées pour ces parties et du manque d'une sémantique claire pour le traitement de ces zones dans les passerelles, en particulier X.400. Cependant, plutôt que de laisser blanche la zone de préambule, de nombreuses

mises en œuvre MIME ont trouvé que c'était un endroit pratique pour insérer une note explicative pour les receveurs qui lisent le message avec un logiciel pré MIME, car de telles notes vont être ignorées par un logiciel conforme à MIME.

Note : Parce que les délimiteurs de frontière ne doivent pas apparaître dans les parties de corps à encapsuler, un agent d'utilisateur doit être vigilant pour choisir une valeur unique de paramètre de frontière. La valeur de paramètre de frontière dans l'exemple ci-dessus pourrait avoir été le résultat d'un algorithme conçu pour produire des délimiteurs de frontière avec une très faible probabilité d'exister déjà dans les données à encapsuler sans avoir à pré examiner les données. D'autres algorithmes pourraient résulter en des délimiteurs de frontière plus "lisibles" pour un receveur avec un vieil agent d'utilisateur, mais exigeraient plus d'attention à la possibilité que le délimiteur de frontière puisse apparaître au début d'une ligne dans la partie encapsulée. La plus simple ligne de délimiteur de frontière possible est quelque chose comme "---", avec une ligne de délimiteur de frontière de clôture de "-----".

Comme exemple très simple, le message multi parties suivant a deux parties, toutes deux en texte pur, l'une d'elle est explicitement typée et l'autre l'est implicitement :

From: Nathaniel Borenstein <nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Date: Sun, 21 mars 1993 23:56:48 -0800 (PST)
Subject: exemple de message
MIME-Version: 1.0
Content-type: multipart/mixed; boundary="simple boundary"

C'est le préambule. Il est à ignorer, bien que ce soit un endroit pratique pour que les agents de composition incluent une note explicative pour les lecteurs non conformes à MIME.

--simple frontière

C'est implicitement un texte de type US-ASCII pur. Il NE se termine PAS par une coupure de ligne.
--simple frontière
Content-type: text/plain; charset=us-ascii

Ceci est explicitement du type texte US-ASCII pur. Il se termine bien par une coupure de ligne.

--simple frontière--

C'est l'épilogue. Il est aussi à ignorer.

L'utilisation d'un type de support de "multipart" dans une partie de corps au sein d'une autre entité "multipart" est explicitement permis. Dans ce cas, pour des raisons évidentes, il faut veiller à s'assurer que chaque entité "multipart" incorporée utilise un délimiteur de frontière différent. Voir dans la RFC 2049 un exemple d'entités "multipart" incorporées.

L'utilisation du type de support "multipart" avec une seule partie de corps peut être utile dans certains contextes, et est explicitement permise.

Note : l'expérience a montré qu'un type de support "multipart" avec une seule partie de corps est utile pour envoyer des types de support non texte. Cela présente l'avantage de fournir le préambule comme endroit pour inclure les instructions de décodage. De plus, un certain nombre de passerelles SMTP déplacent ou retirent les en-têtes MIME, et un décodeur MIME habile peut arriver à bien deviner les frontières de multi parties même en l'absence de l'en-tête Content-Type et par là réussir à décoder le message.

Le seul paramètre global obligatoire pour le type de support "multipart" est le paramètre de frontière, qui consiste en 1 à 70 caractères parmi un ensemble de caractères connu pour être très robuste au travers des passerelles de messagerie, et NE se terminant PAS par une espace. (Si une ligne de délimiteur de frontière apparaît se terminer par une espace, celle-ci doit être présumée avoir été ajoutée par une passerelle, et doit être supprimée.) Il est spécifié de façon formelle par le BNF suivant :

boundary := 0*69<bchars> bcharsnospace

bchars := bcharsnospace / " "

bcharsnospace := DIGIT / ALPHA / "'" / "(" / ")" / "+" / "_" / "," / "-" / "." / "/" / ":" / "=" / "?"

Globalement, le corps d'une entité "multipart" peut être spécifié comme suit :

dash-boundary := "--" boundary
; frontière prise de la valeur du paramètre de frontière du champ Content-Type.

multipart-body := [preamble CRLF]
dash-boundary transport-padding CRLF
body-part *encapsulation
close-delimiter transport-padding
[CRLF epilogue]

transport-padding := *LWSP-char
; Les composeurs NE DOIVENT PAS générer de bourrage de longueur non zéro, mais les receveurs DOIVENT être capables de traiter le bourrage ajouté par le transport du message.

encapsulation := delimiter transport-padding
CRLF body-part

delimiter := CRLF dash-boundary

close-delimiter := delimiter "--"

preamble := discard-text

epilogue := discard-text

discard-text := *(*text CRLF) *text
; Peut être ignoré ou éliminé.

body-part := MIME-part-headers [CRLF *OCTET]
; Les lignes dans une partie de corps ne doivent pas commencer par le dash-boundary spécifié et le délimiteur ne doit pas apparaître dans la partie de corps. Noter que la sémantique de body-part diffère de celle d'un message, comme décrit dans le texte.

OCTET := <toute valeur d'octet de 0 à 255>

IMPORTANT : la libre insertion de linear-white-space et de commentaires RFC 822 entre les éléments montrés dans ce BNF N'EST PAS autorisée car ce BNF ne spécifie pas un champ d'en-tête structuré.

Note : Dans certaines enclaves de transport, les restrictions de la RFC 822 telles que celle qui limite les corps aux caractères US-ASCII imprimables peut n'être pas appliquée. (C'est-à-dire que il peut exister des domaines de transport qui ressemblent au transport standard de messagerie Internet comme spécifié dans la RFC 821 et qui est supposé par la RFC 822, mais sans certaines restrictions.) Le relâchement de ces restrictions devrait être construit comme étendant localement la définition des corps, par exemple pour inclure des octets en dehors de la gamme US-ASCII, pour autant que ces extensions sont supportées par le transport et documentées adéquatement dans le champ d'en-tête Content-Transfer-Encoding. Cependant, en aucun cas il n'est permis que les en-têtes (en-têtes de message ou de partie de corps) contiennent autre chose que des caractères US-ASCII.

Note : Il manque manifestement au type "multipart" une notion de parties de corps structurées, en rapport. Il est recommandé que ceux qui souhaitent fournir des facilités plus structurées ou intégrées de message multi parties devraient définir des sous types de multipart qui soient syntaxiquement identiques mais définissent des relations entre les diverses parties. Par exemple, des sous types de "multipart" pourraient être définis pour inclure une partie distinctive qui ensuite sera utilisée pour spécifier les relations entre les autres parties, se référant probablement à elles par leur champ Content-ID. Les vieilles mises en œuvre ne vont pas reconnaître le nouveau sous type si cette approche est utilisée, mais vont les traiter comme du multipart/mixed et vont donc être capables de montrer à l'utilisateur les parties qui sont reconnues.

5.1.2 Traitement des messages et multi parties incorporés

Le sous type "message/rfc822" défini dans un paragraphe suivant du présent document n'a pas de condition de terminaison autre que de n'avoir plus de données. De même, une entité "multipart" tronquée de façon impropre peut n'avoir pas de

marqueur de frontière de terminaison, et peut cesser de fonctionner à cause de dysfonctionnements du système de messagerie.

Il est essentiel que de telles entités soient traitées correctement quand elles sont elles-mêmes incorporées dans une autre structure "multipart". Il est donc exigé des mises en œuvre MIME qu'elles reconnaissent les marqueurs de frontière de niveau externe à TOUT niveau d'incorporation interne. Il n'est pas suffisant de vérifier seulement le prochain marqueur attendu ou une autre condition de terminaison.

5.1.3 Sous type Mixed

Le sous type "mixed" de "multipart" est destiné à être utilisé quand les parties du corps sont indépendantes et doivent être groupées dans un ordre particulier. Tous les sous types "multipart" qu'une mise en œuvre ne reconnaît pas doivent être traités comme étant du sous type "mixed".

5.1.4 Sous type Alternative

Le type "multipart/alternative" est syntaxiquement identique à "multipart/mixed", mais sa sémantique est différente. En particulier, chaque partie du corps est une version "alternative" des mêmes informations.

Les systèmes devraient reconnaître que le contenu des diverses parties est interchangeable. Les systèmes devraient choisir le "meilleur" type sur la base de l'environnement local et des références, dans certains cas, même par une interaction avec l'utilisateur. Comme avec "multipart/mixed", l'ordre des parties de corps est significatif. Dans ce cas, les solutions de remplacement apparaissent dans l'ordre de fiabilité croissante du contenu original. En général, le meilleur choix est la dernière partie d'un type pris en charge par l'environnement local du système receveur.

"Multipart/alternative" peut être utilisé, par exemple, pour envoyer un message dans un format de texte fantaisiste d'une façon telle qu'il puisse facilement être affiché partout :

```
From: Nathaniel Borenstein <nsb@bellcore.com>
To: Ned Freed <ned@innosoft.com>
Date: Mon, 22 Mar 1993 09:41:09 -0800 (PST)
Subject: message de texte formaté
MIME-Version: 1.0
Content-Type: multipart/alternative; boundary=boundary42

--boundary42
Content-Type: text/plain; charset=us-ascii

... la version en texte pur du message vient ici ...

--boundary42
Content-Type: text/enriched

...La version RFC 1896 text/enriched du même message vient ici ...

--boundary42
Content-Type: application/x-whatever

... la version la plus fantaisiste du même message vient ici ...

--boundary42--
```

Dans cet exemple, les utilisateurs dont le système de messagerie comprend le format "application/x-whatever" vont voir seulement la version fantaisiste, tandis que les autres utilisateurs ne vont voir que la version enrichie ou de texte pur, selon les capacités de leur système.

En général, les agents d'utilisateur qui composent des entités "multipart/alternative" doivent placer les parties de corps en ordre croissant de préférence, c'est-à-dire, avec le format préféré en dernier. Pour le texte fantaisiste, l'agent d'utilisateur envoyeur devrait mettre en premier le format le plus simple et le format le plus riche en dernier. Les agents d'utilisateur receveurs devraient prendre et afficher le dernier format qu'ils sont capables d'afficher. Dans le cas où une des solutions de

remplacement est elle-même de type "multipart" et contient des sous parties non reconnues, l'agent d'utilisateur peut choisir de montrer cette solution de remplacement, une solution antérieure, ou les deux.

Note : Du point de vue de la mise en œuvre, il peut sembler plus normal d'inverser cet ordre et d'avoir la solution la plus pure en dernier. Cependant, la placer en premier est l'option la plus facile pour la lecture quand des entités "multipart/alternative" sont vues à l'aide d'un système non conforme à MIME. Alors que cette approche impose une certaine charge aux système conformes à MIME, l'interopérabilité avec les plus anciens systèmes de messagerie a été réputée plus importante dans ce cas.

Il se peut que certains agents d'utilisateur, si ils peuvent reconnaître plus d'un des formats, préfèrent offrir à l'utilisateur le choix de celui qui sera vu. Cela a un sens, par exemple, si un message inclut à la fois une version d'image bien formatée et une version texte facile à éditer. Ce qui est le plus critique, cependant, est qu'on ne montre pas automatiquement à l'utilisateur les multiples versions des mêmes données. Soit on devrait montrer à l'utilisateur la dernière version reconnue, soit on devrait lui donner le choix.

Sémantique de content-id dans multipart/alternative : chaque partie d'une entité "multipart/alternative" représente les mêmes données, mais les transpositions entre les deux ne sont pas nécessairement sans perte d'information. Par exemple, de l'information est perdue lors de la traduction de ODA en PostScript ou "plain text". Il est recommandé que chaque partie ait une valeur différente de Content-ID dans le cas où le contenu des informations des deux parties n'est pas identique. Et quand le contenu des informations est identique -- par exemple, quand plusieurs parties du type "message/external-body" spécifient des façons différentes d'accéder à des données identiques -- la même valeur de champ Content-ID devrait être utilisée, pour optimiser tous les mécanismes de mise en mémoire tampon qui pourraient être présents du côté du receveur. Cependant, les valeurs de Content-ID utilisées par les parties NE devraient PAS être la même valeur que le Content-ID qui décrit le "multipart/alternative" comme tout, si il y a un tel champ Content-ID. C'est-à-dire, une valeur de Content-ID va se référer à l'entité "multipart/alternative", tandis que une ou plusieurs autres valeurs de Content-ID vont se référer aux parties qu'elle contient.

5.1.5 Sous type Digest

Le présent document définit un sous type "digest" du Content-Type "multipart". Ce type est syntaxiquement identique à "multipart/mixed", mais sa sémantique est différente. En particulier, dans "digest", la valeur de Content-Type par défaut pour une partie de corps est changée de "text/plain" en "message/rfc822". Ceci est fait pour permettre un format de résumé plus lisible qui soit largement compatible (sauf la convention de guillemets) avec la RFC 934.

Note : Bien qu'il soit possible de spécifier une valeur de Content-Type pour une partie de corps dans un résumé qui soit autre que "message/rfc822", comme une partie "text/plain" contenant une description du matériel du résumé, il n'est pas souhaitable de faire ainsi. Le Content-Type "multipart/digest" est destiné à être utilisé pour envoyer des collections de messages. Si une partie "text/plain" est nécessaire, elle devrait être incluse comme partie séparée d'un message "multipart/mixed".

Un résumé dans ce format pourrait alors ressembler à quelque chose comme :

```
From: Moderator-Address
To: Recipient-List
Date: Thue, 22 Mar 1994 13:34:51 +0000
Subject: Internet Digest, volume 42
MIME-Version: 1.0
Content-Type: multipart/mixed;
boundary="---- frontière principale ----"

------ frontière principale ----

...Texte introductif ou table des matières...

------ frontière principale ----
Content-Type: multipart/digest;
boundary="---- prochain message ----"

------ prochain message ----

From: quelqu'un-d'autre
```

Date: Fri, 26 Mar 1993 11:13:32 +0200
Subject: mon opinion

...le corps vient ici ...

------ prochain message ----

From: quelqu'un-d'autre-encore
Date: Fri, 26 Mar 1993 10:07:13 -0500
Subject: mon opinion différente

... un autre corps vient ici ...

------ prochain message ------

------ frontière principale ------

5.1.6 Sous type Parallel

Le présent document définit un sous type "parallel" du type de contenu "multipart". Ce type est syntaxiquement identique à "multipart/mixed", mais sa sémantique est différente. En particulier, dans une entité parallèle, l'ordre des parties de corps n'est pas significatif.

Une présentation courante de ce type est d'afficher toutes les parties simultanément sur le matériel et logiciel qui est capable de le faire. Cependant, les agents de composition devraient savoir que de nombreux lecteurs de messagerie n'ont pas cette capacité et vont montrer les parties à la suite les unes des autres dans tous les cas.

5.1.7 Autres sous types Multipart

D'autres sous types "multipart" sont prévus à l'avenir. Les mises en œuvre MIME doivent en général traiter les sous types non reconnus de "multipart" comme étant équivalents à "multipart/mixed".

5.2 Type de support Message

Il est fréquemment désirable, dans l'envoi de messagerie, d'encapsuler un autre message. Un type spécial de support, "message", est défini pour le faciliter. En particulier, le sous type de "message" "rfc822" est utilisé pour encapsuler les messages RFC 822.

Note : Il a été suggéré que les sous types de "message" pourraient être définis pour transmettre ou rejeter des messages. Cependant, les messages transmis et rejetés peuvent être traités comme des messages multi parties dans lesquels la première partie contient toutes les informations de contrôle ou de description, et une seconde partie, de type "message/rfc822", est le message transmis ou rejeté. Composer un rejet et une transmission de messages de cette manière va préserver les informations de type sur le message original et lui permettre d'être correctement présenté au receveur, et est donc fortement conseillé.

Les sous types de "message" imposent souvent des restrictions sur les codages permis. Ces restrictions sont décrites en conjonction avec chaque sous type spécifique.

Les passerelles de messagerie, les relais, et autres agents de traitement de messagerie sont bien connus pour altérer l'en-tête de niveau supérieur d'un message RFC 822. En particulier, il ajoutent fréquemment, suppriment, ou réarrangent l'ordre des champs d'en-tête. Ces opérations sont explicitement interdites pour les en-têtes encapsulés incorporés dans le corps des messages de type "message."

5.2.1 Sous type RFC822

Un type de support de "message/rfc822" indique que le corps contient un message encapsulé, qui a la syntaxe d'un message RFC 822. Cependant, à la différence des messages RFC 822 de niveau supérieur, la restriction que chaque corps "message/rfc822" doit inclure un en-tête "From", "Date", et au moins un en-tête de destination est supprimée et remplacée par l'exigence qu'au moins un en-tête "From", "Subject", ou "Date" doit être présent.

Il devrait être noté que, en dépit de l'utilisation du numéro "822", une entité "message/rfc822" ne se restreint pas à du matériel en stricte conformité à la RFC822, pas plus que la sémantique des objets "message/rfc822" ne se restreint à la sémantique définie dans la RFC822. Plus précisément, un message "message/rfc822" pourrait être un article de nouvelles ou un message MIME.

Aucun codage autre que "7bit", "8bit", ou "binary" n'est permis pour le corps d'une entité "message/rfc822". Les champs d'en-tête de message sont toujours en US-ASCII dans tous les cas, et les données dans le corps peuvent être codées, et dans ce cas, le champ d'en-tête Content-Transfer-Encoding dans le message encapsulé le reflétera. Du texte non US-ASCII dans les en-têtes d'un message encapsulé peut être spécifié en utilisant les mécanismes décrits dans la RFC 2047.

5.2.2 Sous type Partial

Le sous type "partial" est défini pour permettre que de grandes entités soient livrées comme plusieurs pièces séparées de messagerie et automatiquement réassemblées par un agent d'utilisateur receveur. (Le concept est similaire à celui de la fragmentation et réassemblage IP dans les protocoles Internet de base.) Ce mécanisme peut être utilisé quand des agents de transport intermédiaires limitent la taille des messages individuels qui peuvent être envoyés. Le type de support "message/partial" indique donc que le corps contient un fragment d'une entité plus grande.

Parce que les données de type "message" ne peuvent jamais être codées en base64 ou en quoted-printable, un problème peut se poser si les entités "message/partial" sont construites dans un environnement qui prend en charge le transport binaire ou 8bit. Le problème est que les données binaires vont être partagées en plusieurs messages "message/partial", dont chacun exige un transport binaire. Si de tels messages se rencontrent à une passerelle dans un environnement de transport 7bit, il n'y aura aucun moyen de les coder correctement pour le monde à 7bit, en plus de devoir attendre tous les fragments, rassembler le message interne, et ensuite coder les données réassemblées en base64 ou quoted-printable. Comme il est possible que différents fragments puissent passer par des passerelles différentes, ce n'est pas une solution acceptable. Pour cette raison, il est spécifié que les entités de type "message/partial" doivent toujours avoir un content-transfer-encoding de 7bit (par défaut). En particulier, même dans des environnements qui prennent en charge le transport binaire ou 8bit, l'utilisation d'un content-transfer-encoding de "8bit" ou "binary" est explicitement interdit pour les entités MIME de type "message/partial". Ceci à son tour implique que le message interne ne doit pas utiliser le codage "8bit" ou "binary".

Parce que certains agents de transfert de message peuvent choisir de fragmenter automatiquement les grands messages, et parce que de tels agents peuvent utiliser des seuils de fragmentation très différents, il est possible que les pièces d'un message partiel, puissent se révéler au réassemblage comprendre elles-mêmes un message partiel. Ceci est explicitement permis.

Trois paramètres doivent être spécifiés dans le champ Content-Type de type "message/partial": Le premier, "id", est un identifiant univoque, aussi proche d'un identifiant unique au monde que possible, à utiliser pour confronter les fragments. (En général, l'identifiant est essentiellement un message-id ; si il est placé entre des guillemets, il peut être TOUT message-id, en accord avec le BNF de "parameter" donné dans la RFC 2045.) Le second, "number", un entier, est le numéro de fragment, qui indique où va ce fragment dans la séquence de fragments. Le troisième, "total", un autre entier, est le nombre total de fragments. Ce troisième sous champ est exigé sur le fragment final, et est facultatif (bien que conseillé) sur les fragments antérieurs. Noter aussi que ces paramètres peuvent être dans n'importe quel ordre.

Donc, la seconde pièce d'un message de trois pièces peut avoir l'un ou l'autre des champs d'en-tête suivants

Content-Type: Message/Partial; number=2; total=3; id="oc=jpbe0M2Yt4s@thumper.bellcore.com"

Content-Type: Message/Partial; id="oc=jpbe0M2Yt4s@thumper.bellcore.com"; number=2

Mais la troisième pièce DOIT spécifier le nombre total de fragments :

Content-Type: Message/Partial; number=3; total=3; id="oc=jpbe0M2Yt4s@thumper.bellcore.com"

Noter que le numérotage des fragments commence à 1, pas 0.

Quand les fragments d'une entité fragmentée de cette manière sont mis ensemble, le résultat est toujours une entité MIME complète, qui peut avoir son propre champ d'en-tête Content-Type, et donc peut contenir tout autre type de données.

5.2.2.1 Fragmentation et réassemblage de message

La sémantique d'un message partiel réassemblé doit être celle du message "interne", plutôt que d'un message contenant le message interne. Cela rend possible, par exemple, d'envoyer un grand message audio comme plusieurs messages partiels, et de le faire quand même apparaître au receveur comme un simple message audio plutôt que comme un message encapsulé contenant un message audio. C'est-à-dire que l'encapsulation du message est considérée comme étant "transparente".

Lors de la génération et du réassemblage des pièces d'un message "message/partial", les en-têtes du message encapsulé doivent être fusionnés avec les en-têtes des entités qui l'enclosent. Dans ce processus, les règles suivantes doivent être observées :

(1) Les agents de fragmentation doivent partager les messages aux seules frontières de ligne. Cette restriction est imposée parce que les partages à des points autres que les fins de lignes dépendent de ce que les transports de messages soient capables de préserver la sémantique des messages qui ne se finissent pas sur une séquence CRLF. De nombreux transports sont incapables de préserver une telle sémantique.

(2) Tous les champs d'en-tête provenant du message initial qui enclot, sauf ceux qui commencent par "Content-" et les champs d'en-tête spécifiques "Subject", "Message-ID", "Encrypted", et "MIME-Version", doivent être copiés, dans l'ordre, au nouveau message.

(3) Les champs d'en-tête dans le message enclos qui commencent par "Content-", plus les champs "Subject", "Message-ID", "Encrypted", et "MIME-Version", doivent être ajoutés, dans l'ordre, aux champs d'en-tête du nouveau message. Tous les champs d'en-tête dans le message enclos qui ne commencent pas par "Content-" (sauf les champs "Subject", "Message-ID", "Encrypted", et "MIME-Version") vont être ignorés et éliminés.

(4) Tous les champs d'en-tête provenant du second et de tout message enclosant suivant sont éliminés par le processus de réassemblage.

5.2.2.2 Exemple de fragmentation et réassemblage

Si un message audio est coupé en deux pièces, la première pièce pourrait ressembler à quelque chose comme :

```
X-Weird-Header-1: Foo
From: Bill@host.com
To: joe@otherhost.com
Date: Fri, 26 Mar 1993 12:59:38 -0500 (EST)
Subject: Audio messagerie (partie 1 de 2)
Message-ID: <id1@host.com>
MIME-Version: 1.0
Content-type: message/partial; id="ABC@host.com";

number=1; total=2

X-Weird-Header-1: Bar
X-Weird-Header-2: Hello
Message-ID: <anotherid@foo.com>
Subject: Audio messagerie
MIME-Version: 1.0
Content-type: audio/basic
Content-transfer-encoding: base64

   ... la première moitié des données audio codées vient ici ...
```

et la seconde moitié pourrait ressembler à quelque chose comme ceci :

```
From: Bill@host.com
To: joe@otherhost.com
Date: Fri, 26 Mar 1993 12:59:38 -0500 (EST)
Subject: Audio messagerie (partie 2 de 2)
MIME-Version: 1.0
Message-ID: <id2@host.com>
```

Content-type: message/partial; id="ABC@host.com"; number=2; total=2

... la seconde moitié des données audio codées vient ici ...

Ensuite, quand le message fragmenté est réassemblé, le message résultant à afficher à l'utilisateur devrait ressembler à quelque chose comme ceci :

X-Weird-Header-1: Foo
From: Bill@host.com
To: joe@otherhost.com
Date: Fri, 26 Mar 1993 12:59:38 -0500 (EST)
Message-ID: <unautreid@foo.com>
Subject: Audio messagerie
MIME-Version: 1.0
Content-type: audio/basic
Content-transfer-encoding: base64

... la première moitié des données audio codées vient ici ...
... la seconde moitié des données audio codées vient ici ...

L'inclusion d'un champ "References" dans les en-têtes de la seconde pièce et des suivantes d'un message fragmenté qui fait référence à l'identifiant de message de la pièce précédente peut être avantageuse pour les lecteurs de messagerie qui comprennent et suivent les références. Cependant, la génération de tels champs "References" est entièrement facultative.

Finalement, on devrait noter que le champ d'en-tête "Encrypted" a été rendu obsolète par la messagerie à confidentialité améliorée (PEM, *Privacy Enhanced Messaging*) [RFC1421], [RFC1422], [RFC1423], [RFC1424], mais les règles ci-dessus sont néanmoins supposées décrire la façon correcte de le traiter si il est rencontré dans le contexte d'une conversion de, et en, des fragments de "message/partial".

5.2.3 Sous type External-Body

Le sous type "external-body" indique que les données de corps réelles ne sont pas incluses, mais simplement référencées. Dans ce cas, les paramètres décrivent un mécanisme pour accéder aux données externes.

Quand une entité MIME est du type "message/external-body", elle comporte un en-tête, deux CRLF consécutifs, et l'en-tête de message pour le message encapsulé. Si une autre paire de CRLF consécutifs apparaît, cela termine bien sûr l'en-tête de message pour le message encapsulé. Cependant, comme le corps du message encapsulé est lui-même externe, il N'apparaît PAS dans la zone qui suit. Par exemple, considérons le message suivant :

Content-type: message/external-body;
access-type=local-file;
name="/u/nsb/Me.jpeg"

Content-type: image/jpeg
Content-ID: <id42@guppylake.bellcore.com>
Content-Transfer-Encoding: binary

CECI N'EST PAS RÉELLEMENT LE CORPS !

La zone à la fin, qui pourrait être appelée le "corps fantôme", est ignorée pour la plupart des messages "external-body". Cependant, elle peut être utilisée pour contenir des informations auxiliaires pour certains de ces messages, comme c'est le cas quand le type d'accès est "mail-server". Le seul type d'accès défini dans le présent document qui utilise le corps fantôme est "mail-server", mais d'autres types d'accès pourront être définis à l'avenir dans d'autres spécifications qui utilisent cette zone.

Les en-têtes encapsulés dans TOUTES les entités "message/external-body" DOIVENT inclure un champ d'en-tête Content-ID pour donner un identifiant univoque par lequel faire référence aux données. Cet identifiant peut être utilisé pour des mécanismes de mise en antémémoire, et pour reconnaître la réception des données quand le type d'accès est "mail-server".

Noter que, comme spécifié ici, il est exigé que les jetons qui décrivent les données de corps externe, comme les noms de fichier et les commandes de serveur de messagerie soient dans le jeu de caractères US-ASCII.

Si cela se révèle problématique en pratique, un nouveau mécanisme peut être requis comme future extension à MIME, soit comme de nouveaux types d'accès définis pour "message/external-body", soit par un autre mécanisme.

Comme avec "message/partial", les entités MIME de type "message/external-body" DOIVENT avoir un content-transfer-encoding de 7bit (par défaut). En particulier, même dans les environnements qui prennent en charge le transport binaire ou 8bit, l'utilisation d'un content-transfer-encoding de "8bit" ou "binary" est explicitement interdit pour les entités de type "message/external-body".

5.2.3.1 Paramètres généraux de External-Body

Les paramètres qui peuvent être utilisés avec tout "message/external-body" sont :

(1) ACCESS-TYPE -- mot qui indique le mécanisme d'accès pris en charge, par lequel le fichier ou les données peuvent être obtenus. Ce mot n'est pas sensible à la casse. Les valeurs incluent, sans s'y limiter, "FTP", "ANON-FTP", "TFTP", "LOCAL-FILE", et "MAIL-SERVER". Les valeurs futures, sauf les valeurs expérimentales commençant par "X-", doivent être enregistrées auprès de l'IANA, comme décrit dans la RFC 2048. Ce paramètre est inconditionnellement obligatoire et DOIT être présent sur TOUT "message/external-body".

(2) EXPIRATION -- date (dans la syntaxe "date-time" de la RFC0822, comme étendue par la RFC1123 pour permettre 4 chiffres dans le champ de l'année) après laquelle l'existence des données externes n'est plus garantie. Ce paramètre peut être utilisé avec TOUT type d'accès et est TOUJOURS facultatif.

(3) SIZE -- taille (en octets) des données. L'intention de ce paramètre est d'aider le receveur à décider d'étendre ou non les ressources nécessaires pour restituer les données externes. Noter que ceci décrit la taille des données dans leur forme canonique, c'est-à-dire, avant l'application de tout Content-Transfer-Encoding ou après que les données ont été décodées. Ce paramètre peut être utilisé avec TOUT type d'accès et est TOUJOURS facultatif.

(4) PERMISSION -- champ insensible à la casse qui indique si on s'attend ou non à ce que les clients puissent aussi tenter d'écraser les données. Par défaut, ou si permission est "read" *(lecture)*, l'hypothèse est qu'ils ne sont pas autorisés, et que si les données sont restituées une fois, elles ne sont jamais nécessaires à nouveau. Si PERMISSION est "read-write" *(lecture-écriture)*, cette hypothèse est invalide, et toute copie locale doit être considérée comme une simple antémémoire. "Read" et "Read-write" sont les seules valeurs définies de permission. Ce paramètre peut être utilisé avec TOUT type d'accès et est TOUJOURS facultatif.

La sémantique précise des types d'accès définis ici est décrite dans le paragraphe suivant.

5.2.3.2 Types d'accès "ftp" et "tftp"

Un type d'accès de FTP ou TFTP indique que le corps de message est accessible comme fichier en utilisant respectivement les protocoles FTP [RFC0959] ou TFTP [RFC0783]. Pour ces types d'accès, Les paramètres supplémentaires suivants sont obligatoires :

(1) NAME -- le nom du fichier qui contient les données de corps réelles.

(2) SITE -- une machine dont le fichier peut être obtenu, en utilisant le protocole donné. Ce doit être un nom de domaine pleinement qualifié, pas un surnom.

(3)Avant que toutes données soient restituées, en utilisant FTP, l'utilisateur va généralement avoir besoin qu'on lui demande de fournir un identifiant de connexion et un mot de passe pour la machine désignée par le paramètre du site. Pour des raisons de sécurité, un tel identifiant et un tel mot de passe ne sont pas spécifiés comme paramètres de type de contenu, mais doivent être obtenus de l'utilisateur.

De plus, les paramètres suivants sont facultatifs :

(1) DIRECTORY -- un répertoire duquel les données désignées par NAME devraient être restituées.

(2) MODE -- une chaîne insensible à la casse qui indique le mode à utiliser lors de la restitution des informations. Les valeurs valides pour le type d'accès "TFTP" sont "NETASCII", "OCTET", et "MAIL", comme spécifié par le protocole TFTP [RFC0783]. Les valeurs valides pour le type d'accès "FTP" sont "ASCII", "EBCDIC", "IMAGE", et "LOCALn"

où "n" est un entier décimal, normalement 8. Cela correspond aux types de représentation "A" "E" "I" et "L n" comme spécifié par le protocole FTP [RFC0959]. Noter que "BINARY" et "TENEX" ne sont pas des valeurs valides pour MODE et que "OCTET" ou "IMAGE" ou "LOCAL8" devraient être utilisées à la place. Si MODE n'est pas spécifié, la valeur par défaut est "NETASCII" pour TFTP et "ASCII" autrement.

5.2.3.3 Types d'accès "anon-ftp"

Le type d'accès "anon-ftp" est identique au type d'accès "ftp", sauf que l'utilisateur n'a pas besoin qu'on lui demande de fournir un nom et un mot de passe pour le site spécifié. À la place, le protocole ftp va être utilisé avec la connexion "anonymous" et un mot de passe qui correspond à l'adresse de messagerie de l'utilisateur.

5.2.3.4 Types d'accès "local-file"

Un type d'accès de "local-file" indique que le corps réel est accessible comme fichier sur la machine locale. Deux paramètres supplémentaires sont définis pour ce type d'accès :

(1) NAME -- nom du fichier qui contient les données du corps réel. Ce paramètre est obligatoire pour le type d'accès "local-file".

(2) SITE -- un spécificateur de domaine pour une machine ou ensemble de machines qui sont connues pour avoir accès au fichier de données. Ce paramètre facultatif est utilisé pour décrire la localisation de référence pour les données, c'est-à-dire le ou les sites auxquels le fichier est supposé être visible. Des astérisques peuvent être utilisés pour rechercher la correspondance par un caractère générique à une partie d'un nom de domaine, comme "*.bellcore.com", pour indiquer un ensemble de machines sur lesquelles les données devraient être directement visibles, tandis qu'un seul astérisque peut être utilisé pour indiquer un fichier qui est supposé être universellement disponible, par exemple, via un système de fichiers mondial.

5.2.3.5 Types d'accès "mail-server"

Le type d'accès "mail-server" indique que le corps réel est disponible à partir d'un serveur de messagerie. Deux paramètres supplémentaires sont définis pour ce type d'accès :

(1) SERVER -- la spécification de l'adresse du serveur de messagerie d'où les données du corps réel peuvent être obtenues. Ce paramètre est obligatoire pour le type d'accès de "mail-server".

(2) SUBJECT -- le sujet qui va être utilisé dans le message qui est envoyé pour obtenir les données. Noter que les serveurs de messagerie chiffrée sur les lignes "Subject" NE sont PAS recommandés, mais il est connu que de tels serveurs de messagerie existent. C'est un paramètre facultatif.

Parce que les serveurs de messagerie acceptent des syntaxes diverses, dont certaines sont multi lignes, la commande complète à envoyer à un serveur de messagerie n'est pas incluse comme paramètre dans le champ d'en-tête content-type. À la place, il est fourni comme "corps fantôme" quand le type de support est "message/external-body" et le type d'accès est "mail-server".

Noter que MIME ne définit pas de syntaxe de serveur de messagerie. Il permet plutôt l'inclusion de commandes arbitraires de serveur de messagerie dans le corps fantôme. Les mises en œuvre doivent inclure le corps fantôme dans le corps du message qu'elles envoient à l'adresse du serveur de messagerie pour restituer les données pertinentes.

À la différence des autres types d'accès, l'accès "mail-server" est asynchrone et va se faire dans un délai imprévisible dans l'avenir. Pour cette raison, il est important qu'il y ait un mécanisme par lequel les données retournées puissent être confrontées à l'entité originale "message/external-body". Les serveurs de messagerie MIME doivent utiliser le même champ Content-ID sur le message retourné que utilisé dans les entités originales "message/external-body", pour faciliter une telle confrontation.

5.2.3.6 Problèmes de sécurité de External-Body

Les entités "message/external-body" font apparaître deux importants problèmes de sécurité :

(1) Accéder aux données via une référence "message/external-body" résulte effectivement en ce que le receveur du

message effectue une opération qui a été spécifiée par le générateur du message. Il est donc possible que le générateur du message trompe un receveur en lui faisant faire quelque chose qu'il n'aurait pas fait autrement. Par exemple, un générateur pourrait spécifier une action qui tente la restitution de matériel que le receveur n'est pas autorisé à obtenir, causant la violation involontaire par le receveur d'une politique de sécurité. Pour cette raison, les agents d'utilisateur capables de résoudre des références externes doivent toujours prendre des mesures pour décrire au receveur l'action qu'ils vont entreprendre et demander explicitement la permission avant de l'effectuer.

Le type d'accès "mail-server" est particulièrement vulnérable, en ce qu'il cause l'envoi par le receveur d'un nouveau message dont le contenu est spécifié par le générateur du message d'origine. Étant donné le potentiel d'abus, tout message de demande qui est construit de cette sorte devrait contenir une claire indication qu'il a été généré automatiquement (par exemple dans un champ d'en-tête Comments:) pour tenter de résoudre une référence MIME "message/external-body".

(2) MIME va parfois être utilisé dans des environnements qui fournissent des garanties d'intégrité et d'authenticité du message. Si elles sont présentes, de telles garanties ne peuvent s'appliquer qu'au contenu direct réel des messages -- elles peuvent ou non s'appliquer aux données auxquelles on accède par le mécanisme "message/external-body" de MIME. En particulier, il est possible de subvertir certains mécanismes d'accès même quand le système de messagerie est lui-même sûr.

On devrait noter que ce problème existe avec ou sans la disponibilité des mécanismes MIME. Une référence fortuite à un site FTP contenant un document dans le texte d'un message sûr pose des problèmes similaires -- la seule différence est que MIME assure la restitution automatique d'un tel matériel, et les utilisateurs peuvent faire une confiance non garantie à de tels mécanismes de restitution automatique.

5.2.3.7 Exemples et autres explications

Quand le mécanisme "external-body" est utilisé en conjonction avec le type de support "multipart/alternative" il étend la fonctionnalité de "multipart/alternative" pour inclure le cas où la même entité est fournie dans le même format mais via des mécanismes d'accès différents. Quand ceci est fait, l'origine du message doit ordonner les parties d'abord en termes de formats préférés et ensuite par mécanismes d'accès préféré. Le logiciel de restitution du receveur devrait alors évaluer la liste en termes de format et en termes de mécanismes d'accès.

Avec la possibilité émergente de systèmes de fichiers de très large zone, il devient très difficile de savoir à l'avance l'ensemble de machines où un fichier sera ou non accessible directement à partir du système de fichiers. Donc il peut y avoir du sens à fournir à la fois un nom de fichier, à essayer directement, et le nom d'un ou plusieurs sites à partir desquels il est connu que le fichier est accessible. Une mise en œuvre peut essayer de restituer des fichiers distants en utilisant FTP ou tout autre protocole, en utilisant la restitution de fichier anonyme ou en invitant l'utilisateur sur le nom et mot de passe nécessaires. Si un corps externe est accessible via plusieurs mécanismes, l'envoyeur peut inclure plusieurs entités de type "message/external-body" dans les parties de corps d'une entité "multipart/alternative" enclosante

Cependant, le mécanisme "external-body" n'est pas destiné à se limiter à la restitution de fichier, comme le montre le type d'accès "mail-server". Au delà de cela, on peut imaginer, par exemple, d'utiliser un serveur vidéo pour des références externes à des extraits de vidéo.

Les champs d'en-tête de message incorporé qui apparaissent dans le corps de données "message/external-body" doivent être utilisés pour déclarer le type de support du corps externe si il est autre chose que du pur texte US-ASCII, car le corps externe n'a pas de section d'en-tête pour déclarer son type. De même, tout content-transfer-encoding autre que "7bit" doit aussi être déclaré ici. Donc un message "message/external-body" complet, se référant à un objet en format PostScript, pourrait ressembler à :

```
From: N'importe qui
To: Quelqu'un
Date: N'importe quand
Subject: N'importe quoi
MIME-Version: 1.0
Message-ID: <id1@host.com>
Content-Type: multipart/alternative; boundary=42
Content-ID: <id001@guppylake.bellcore.com>

--42
Content-Type: message/external-body; name="BodyFormats.ps";
```

```
        site="thumper.bellcore.com"; mode="image";
        access-type=ANON-FTP; directory="pub";
        expiration="Fri, 14 Jun 1991 19:13:14 -0400 (EDT)"

Content-type: application/postscript
Content-ID: <id42@guppylake.bellcore.com>

--42
Content-Type: message/external-body; access-type=local-file;
        name="/u/nsb/writing/rfcs/RFC-MIME.ps";
        site="thumper.bellcore.com";
        expiration="Fri, 14 Jun 1991 19:13:14 -0400 (EDT)"

Content-type: application/postscript
Content-ID: <id42@guppylake.bellcore.com>

--42
Content-Type: message/external-body;
         access-type=mail-server;
         server="listserv@bogus.bitnet";
         expiration="Fri, 14 Jun 1991 19:13:14 -0400 (EDT)"

Content-type: application/postscript
Content-ID: <id42@guppylake.bellcore.com>

obtenir RFC-MIME.DOC

--42--
```

Noter que dans les exemples ci-dessus, le content-transfer-encoding par défaut de "7bit" est supposé pour les données postscript externes.

Comme le type "message/partial", le type de support "message/external-body" est destiné à être transparent, c'est-à-dire, à porter le type de données dans le corps externe plutôt que de porter un message avec un corps de ce type. Donc, les en-têtes sur les parties externes et internes doivent être fusionnées en utilisant les mêmes règles que pour "message/partial". En particulier, cela signifie que les champs Content-type et Subject sont écrasés, mais que le champ From est préservé.

Noter que comme les corps externes ne sont pas transportés avec les références du corps externe, ils n'ont pas besoin de se conformer aux limitations de transport qui s'appliquent aux références elles-mêmes. En particulier, les transports de messagerie Internet peuvent imposer le 7bit et les limites de longueur de ligne, mais celles-ci ne s'appliquent pas automatiquement aux références de corps externe binaire. Donc, un Content-Transfer-Encoding n'est généralement pas nécessaire, bien que permis.

Noter que le corps d'un message de type "message/external-body" est gouverné par la syntaxe de base d'un message de la RFC0822. En particulier, tout ce qui est avant la première paire de CRLF consécutifs sont des informations d'en-tête, tandis que tout ce qui est après sont des informations de corps, ce qui est ignoré pour la plupart des types d'accès.

5.2.4 Autres sous types de message

Les mises en œuvre MIME doivent en général traiter les sous types non reconnus de "message" comme équivalents à "application/octet-stream".

Les futurs sous types de "message" destinés à être utilisés avec la messagerie électronique devraient être restreints au codage "7bit". Un type autre que "message" devrait être utilisé si la restriction à "7bit" n'est pas possible.

6. Valeurs expérimentales de type de support

Une valeur de type de support commençant par les caractères "X-" est une valeur privée, à utiliser par des systèmes consentants sur accord mutuel. Tout format qui n'a pas une définition rigoureuse et publique doit être nommé avec un

préfixe "X-", et les valeurs spécifiées publiquement ne devront jamais commencer par "X-". (De plus anciennes versions du système largement utilisé Andrew utilisent le nom "X-BE2", de sorte que les nouveaux systèmes devraient probablement choisir un nom différent.)

En général, l'utilisation de types "X-" de niveau supérieur est fortement déconseillée. Les mises en œuvre devraient inventer des sous types des types existants chaque fois que possible. Dans de nombreux cas, un sous type de "application" va être plus approprié qu'un nouveau type de niveau supérieur.

7. Résumé

Les cinq types de support séparés fournissent un mécanisme normalisé pour étiqueter les entités comme "audio", "image", ou plusieurs autres sortes de données.Les types de support composites "multipart" et "message" permettent de mélanger et des structurer hiérarchiquement les entités de différents types dans un seul message. Une syntaxe de paramètres distinctifs permet une spécification plus poussée des détails du format des données, en particulier la spécification de jeux de caractères de remplacement. Des champs d'en-tête facultatifs supplémentaires fournissent des mécanismes pour certaines extensions réputées désirables par de nombreuses mises en œuvre. Finalement, un certain nombre de types de support utiles sont définis pour une utilisation générale par des agents d'utilisateurs consentants, notamment "message/partial" et "message/external-body".

8. Considérations sur la sécurité

Les questions de sécurité sont discutées dans le contexte du type "application/postscript", du type "message/external-body", et dans la RFC2048. Les mises en œuvre devraient porter une attention particulière aux implications pour la sécurité de tous les types de support qui peuvent causer l'exécution à distance de toutes actions dans l'environnement du receveur. Dans de tels cas, la discussion du type "application/postscript" peut servir de modèle pour envisager d'autres types de support avec des capacités d'exécution à distance.

9. Adresse des auteurs

Pour plus d'informations, il vaut mieux contacter les auteurs de ce document par messagerie Internet :

Ned Freed
Innosoft International, Inc.
1050 East Garvey Avenue South
West Covina, CA 91790
USA
téléphone : +1 818 919 3600
fax : +1 818 919 3614
mél : ned@innosoft.com

Nathaniel S. Borenstein
First Virtual Holdings
25 Washington Avenue
Morristown, NJ 07960
USA
téléphone : +1 201 540 8967
fax : +1 201 993 3032
mél : nsb@nsb.fv.com

MIME est le résultat du travail du groupe de travail sur les extensions à la RFC0822 de l'équipe d'ingénierie de l'Internet. On peut contacter le président de ce groupe, Greg Vaudreuil, à :

Gregory M. Vaudreuil
Octel Network Services
17080 Dallas Parkway
Dallas, TX 75248-1905

USA
mél : Greg.Vaudreuil@Octel.Com

Appendice A -- Récapitulation de la grammaire

Cet appendice contient la grammaire BNF complète pour toute la syntaxe spécifiée dans ce document.

Par elle-même, cependant, cette grammaire est incomplète. Elle se réfère par nom à plusieurs règles de syntaxe qui sont définies par la RFC 822. Plutôt que de reproduire ces définitions ici, et risquer des différences involontaires entre les deux, ce document renvoie simplement le lecteur à la RFC 822 pour les définitions restantes. Chaque fois qu'un terme n'est pas défini, il se réfère à la définition de la RFC 822.

boundary := 0*69<bchars> bcharsnospace

bchars := bcharsnospace / " "

bcharsnospace := DIGIT / ALPHA / "'" / "(" / ")" / "+" / "_" / "," / "-" / "." / "/" / ":" / "=" / "?"

body-part := <"message" comme défini dans la RFC 822, avec tous les champs d'en-tête facultatifs, ne commençant pas par la limite spécifiée par un tiret, et avec le délimiteur ne se produisant bulle part dans la partie de corps. Noter que la sémantique d'une partie diffère de la sémantique d'un message, comme décrit dans le texte.>

close-delimiter := delimiter "--"

dash-boundary := "--" boundary
; frontière tirée de la valeur du paramètre de frontière du champ Content-Type.

delimiter := CRLF dash-boundary

discard-text := *(*text CRLF) *text ; Peut être ignoré ou éliminé.

encapsulation := delimiter transport-padding
CRLF body-part

epilogue := discard-text

multipart-body := [preamble CRLF]
dash-boundary transport-padding CRLF
body-part *encapsulation
close-delimiter transport-padding
[CRLF epilogue]

preamble := discard-text

transport-padding := *LWSP-char
; Les composeurs NE DOIVENT PAS générer de bourrage de transport de longueur non zéro , mais les receveurs DOIVENT être capables de traiter le bourrage ajouté par les transports de message.