Groupe de travail Réseau
Request for Comments : 5198
RFC rendue obsolète : 0698
RFC mise à jour : 0854
Catégorie : Sur la voie de la normalisation

J. Klensin
M. Padlipsky
mars 2008

Traduction Claude Brière de L'Isle

Format Unicode pour les échanges du réseau

Résumé
L'Internet d'aujourd'hui a besoin d'une forme normalisée pour la transmission des informations de "texte" internationalisées, en parallèle aux spécifications pour l'utilisation de ASCII qui datent des premier jours de l'ARPANET. Le présent document spécifie ce format, utilisant UTF-8 avec normalisation et des séquences de fin de ligne spécifiques.

Table des matières

1. Introduction

1.1 Exigence d'un format de flux de texte normalisé

Historiquement, les protocoles de l'Internet ont été largement fondés sur l'ASCII et les références à du "texte" dans les protocoles ont supposé que le texte ASCII et spécifiquement le texte en format de terminal virtuel du réseau (NVT, *Network Virtual Terminal*ou "ASCII du réseau" (voir l'Appendice A et l'Appendice B). Les protocoles et formats qui sont passés au delà de l'ASCII ont inclus des arrangements pour identifier spécifiquement le jeu de caractères et souvent le langage utilisé.

Dans notre monde plus internationalisé, "texte" n'est clairement plus égal sans ambiguïté au "ASCII du réseau". Heureusement cependant, on converge sur Unicode [Unicode] [ISO10646] comme seul codage international de caractères d'échange et on n'a plus besoin de traiter des standard par script pour les jeux de caractères (par exemple, une norme pour chaque arabique, cyrillique, devanagari, etc., ou même des normes fixées sur des langages qui sont usuellement considérés partager un script, comme le français, l'allemand ou le suédois). Malheureusement, il est certainement temps cependant de définir un type de texte fondé sur Unicode à utiliser comme format commun d'échange de texte, "utiliser Unicode"

implique même plus d'ambiguïtés que "utiliser ASCII" ne faisait il y a des dizaines d'années.

Unicode identifie chaque caractère par un entier, appelé son "codet", dans la gamme de 0 à 0x10ffff. Ces entiers peuvent être codés en séquences d'octets pour la transmission dans au moins trois normes et formes de codages généralement reconnues, dont toutes sont complètement définies dans la norme Unicode et les documents cités ci-dessous :

- UTF-8 [RFC3629] définit un codage de longueur variable qui peut être appliqué uniformément à tous les codets.

- UTF-16 [RFC2781] code la gamme de caractères Unicode dont les codets sont moins de 65536 directement comme des entiers de 16 bits, et fournit un mécanisme de "substitution" pour coder les codets supérieurs dans 32 bits.

- UTF-32 (aussi appelé UCS-4) code simplement chaque codet comme un entier de 32 bits.

Des formes et nomenclatures plus anciennes, comme l'UCS-2 à 16 bits, sont maintenant fortement déconseillées.

Comme avec l'ASCII, chacune de ces formes peut être utilisée avec différentes conventions de terminaison de ligne. Cette souplesse peut être une source supplémentaire de confusion avec, par exemple, des références d'indice (décalages) dans les documents fondés sur le compte de caractères.

Le présent document propose d'établir un "Net-Unicode" comme nouvelle forme normalisée de transmission de texte pour l'Internet, pour servir de solution de remplacement internationalisée à l'ASCII NVT quand il est spécifié dans les protocoles nouveaux -- et, lorsque approprié, mis à jour. UTF-8 [RFC3629] est choisi pour le codage parce que il a de bonnes propriétés de compatibilité avec l'ASCII et pour d'autres raisons discutées dans la politique existante de jeu de caractères de l'IETF [RFC2277]. "Net-Unicode" est spécifié à la Section 2 ; les sections suivantes du document donnent les fondements et les explications.

Chaque fois qu'il y a un choix, Unicode DEVRAIT être utilisé avec le codage de texte spécifié ici. Cette combinaison est préférée au codage sur deux octets de "ASCII étendu" [RFC0698] ou les systèmes de codages de caractères assortis par langue ou par-pays.

1.2 Terminologie

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

2. Définition de Net-Unicode

Le format Unicode du réseau (Net-Unicode) est défini comme suit. Des parties de cette définition sont délibérément informelles, donnant des lignes directrices pour des profils ou règles spécifiques dans les protocoles qui y font référence plutôt que des règles fermes qui s'appliquent partout.

1. Les caractères DOIVENT être codés en UTF-8 comme défini dans la [RFC3629].

2. Si le protocole a le concept de "lignes", les terminaisons de ligne DOIVENT être indiquées par la séquence Retour chariot (CR, U+000D) suivie par Saut à la ligne (LF, *Line-Feed*) (U+000A) souvent appelé juste CRLF. Un CR NE DEVRAIT PAS apparaître sauf quand il est suivi d'un LF. Le seul autre contexte dans lequel un CR est permis est dans la combinaison CR NUL, qui n'est pas recommandée (voir la note à la fin de ce paragraphe).

3. Les caractères de contrôle dans la gamme ASCII (de U+0000 à U+001F et de U+007F à U+009F) DEVRAIENT généralement être évités. Espace (SP, U+0020), CR, LF, et saut à la page (FF, *Form Feed*) (U+000C) sont des exceptions à ce principe, mais l'utilisation de tous sauf le premier exige des précautions comme discuté plus loin. Les "contrôles C1" (U+0080 à U+009F) qui n'apparaissent pas en ASCII, NE DOIVENT PAS apparaître.

   FF devrait n'être utilisé qu'avec prudence : il n'a pas une interprétation standard et universelle et, en particulier, si son utilisation suppose une longueur de page, une telle hypothèse peut n'être pas appropriée dans un contexte international (par exemple, si on considère une feuille de 8,5x11 pouces par rapport à A4). D'autres caractères de contrôle sont utilisés pour affecter le format d'affichage, les appareils de contrôle, ou pour structurer les fichiers. Aucune de ces

utilisation n'est appropriée pour les flux de texte.

4. Avant la transmission, toutes les séquences de caractères DEVRAIENT être normalisées conformément à la forme de normalisation Unicode "NFC" (voir la Section 3).

5. Comme suggéré à la Section 6 de la RFC 3629, la signature de marque d'ordre des octets (BOM, *Byte Order Mark*) NE DOIT PAS apparaître au début de ces chaînes de texte.

6. Les systèmes conformes à la présente spécification NE DOIVENT PAS transmettre de chaîne contenant un codet non alloué dans la version de Unicode dont ils dépendent. La version de NFC et la version de Unicode utilisées par ce système DOIVENT être cohérentes.

L'utilisation de LF sans CR est discutable ; voir à l'Appendice B. Les caractères de contrôle plus récents IND (U+0084) et NEL ("Next Line", U+0085) auraient pu être utilisés pour désambigüer les diverses situations de terminaison de ligne, mais parce que leur utilisation n'a pas été bien établie dans l'Internet, parce que de nombreux protocoles exigent le CRLF, et parce que IND et NEL tombent dans le groupe des "Contrôles C1" (voir ci-dessous) ils NE DOIVENT PAS être utilisés. Des observations similaires s'appliquent aux encore plus nouveaux séparateurs de lignes et paragraphes à U+2028 et U+2029 et à tout futur caractère qui pourrait être défini pour servir ces fonctions. Pour la présente spécification et les protocoles qui dépendent d'elle, les lignes se terminent par CRLF et seulement CRLF. Tous ce qui ne se termine par par un CRLF soit n'est pas une ligne, soit est sévèrement mal formé.

La spécification NVT contenait un certain nombre de dispositions supplémentaires, par exemple, pour l'utilisation facultative de l'espace arrière et du "CR nu" (envoyé comme CR NUL) pour générer des séquences de caractères surchargés. Le bien plus grand nombre de caractères précomposés dans Unicode, la disponibilité de caractères de combinaison, et l'usage croissant de conventions de balisage de divers types pour montrer, par exemple, l'emphase (plutôt que de tenter de le faire via l'utilisation de caractères spéciaux) devrait rendre de telles séquences largement inutiles. Ces séquences DEVRAIENT être évitées si c'est possible. Cependant, parce que elles étaient facultatives dans les applications NVT et que la présente spécification est un sur-ensemble de NVT, elles ne peuvent pas être complètement interdites. Le plus important de ces règles est que le CR NE DOIT PAS apparaître sauf immédiatement suivi par LF (indiquant une fin de ligne) ou NUL. Parce que NUL (octet dont la valeur est toute de zéros, c'est-à-dire, %x00 dans la notation de la [RFC5234]) est hostile aux langages de programmation qui utilisent ce caractère comme délimiteur de chaîne, la séquence CR NUL DEVRAIT être aussi évitée pour cette raison.

3. Normalisation

Il y des cas où les chaînes de Unicode sont fondamentalement équivalentes, représentant essentiellement le même texte. Elles sont appelées des "équivalents canoniques" dans la norme Unicode. Par exemple, les paires de chaînes suivantes sont canoniquement équivalentes :

U+2126 OHM SIGN Ω
U+03A9 GREEK CAPITAL LETTER OMEGA

U+0061 LATIN SMALL LETTER A, U+0300 COMBINING GRAVE ACCENT
U+00E0 LATIN SMALL LETTER A WITH GRAVE à

La comparaison des chaînes devient beaucoup plus facile si ces cas sont toujours représentés par une seule forme unique. Le Consortium Unicode spécifie une forme de normalisation, connue sous le nom de NFC [NFC], qui fournit les transpositions et mécanismes nécessaires pour convertir toutes les séquences canoniquement équivalentes en une seule forme unique. Normalement, cette forme produit des caractères précomposés pour toute séquence qui peut être représentée de cette façon. Elle réordonne aussi les autres marques de combinaison afin d'avoir un ordre unique et sans ambiguïté.

Des diverses formes de normalisation définies au titre de Unicode, NFC est la plus proche de l'utilisation pratique actuelle, qui minimise les effets collatéraux du fait de la prise en compte des équivalences de caractères qui peuvent n'être pas équivalentes dans toutes les situations, et exige normalement le moins de travail quand on convertit à partir de codages non-Unicode.

La section précédente exige que, sauf dans des circonstances très inhabituelles, toutes les chaînes Net-Unicode soient transmises en forme normalisée. La reconnaissance du fait que certaines mises en œuvre des applications peuvent s'appuyer sur des bibliothèques de système d'exploitation sur lesquelles elles ont peu de contrôle et l'adhésion au principe

de robustesse suggère que les receveurs de telles chaînes devraient être prêts à en recevoir de non normalisées et de ne pas réagir à cela de façon excessive.

4. Versions de Unicode

Unicode change et s'étend au fil du temps. De larges blocs d'espace sont réservés pour une future expansion. Les nouvelles versions, qui apparaissent à intervalles réguliers, ajoutent de nouveaux scripts et caractères. Occasionnellement elles changent aussi certaines définitions de propriétés. Rétrospectivement, un des avantages de l'ASCII [ASCII] quand il a été choisi était que l'espace de code était plein la première fois que la norme a été publiée. Il n'y avait pas moyen d'ajouter de caractères ou de changer les allocations de codets sans être évidemment incompatible.

Bien qu'il y ait quelques problèmes de sécurité si des gens essayent délibérément de biaiser le système (voir la Section 6) les changements de version Unicode ne devraient pas avoir d'impact significatif sur la spécification des flux de texte du présent document pour les raisons suivantes :

- La transformation entre les positions de tableau de code Unicode et le code UTF-8 correspondant est algorithmique ; elle ne dépend pas de si un codet a été alloué ou non.

- La normalisation recommandée ici, NFC (voir la Section 3) effectue un ensemble très limité de transpositions, beaucoup plus limité que celui du très extensif NFKC utilisé dans, par exemple, Nameprep [RFC3491].

Les tableaux de NFC peuvent être mis à jour au fil du temps lorsque de nouveaux caractères sont ajoutés, mais le Consortium Unicode a garanti la stabilité de toutes les chaînes NFC. C'est-à-dire, si une chaîne ne contient aucun caractère non alloué, et est normalisée conformément à NFC, elle va toujours être normalisée en accord avec toutes les futures versions de la norme Unicode. La stabilité du format Net-Unicode est donc garanti quand une mise en œuvre qui convertit du texte en format Net-Unicode ne permet pas de caractères non alloués.

Parce que les codets Unicode qui sont réservés pour utilisation privée n'ont pas de définitions standard ou d'interprétations de normalisation, ils DEVRAIENT être évités dans les chaînes destinées à des échanges sur l'Internet.

Si Unicode devait être changé d'une façon qui violerait ces hypothèses, c'est-à-dire, qui invaliderait l'ordre de la chaîne d'octets spécifié dans la RFC 3629 ou qui changerait la stabilité de NFC comme indiqué ci-dessus, la présente spécification ne s'appliquerait pas. Dit différemment, la présente spécification s'applique seulement aux versions de Unicode qui commencent à la version 5.0 et s'étendent, sans les inclure, à toute version pour laquelle ces changements sont faits dans la définition de UTF-8 ou à la stabilité de NFC. De tels changements violeraient les politiques établies de Unicode et sont donc improbables, mais, si elles devaient se produire, il serait nécessaire d'évaluer leur compatibilité avec la présente spécification et les autres utilisations de NFC dans l'Internet.

Si la spécification d'un protocole fait référence à la présente, les chaînes qui sont reçues par ce protocole et qui apparaissent être de l'UTF-8 et ne sont pas autrement identifiées (par exemple, par l'étiquetage du jeu de caractères) DEVRAIENT être traitées comme utilisant UTF-8 en conformité avec la présente spécification.

5. Applicabilité et stabilité de la spécification

5.1 Utilisation dans les spécifications d'applications de l'IETF

Durant le développement de la présente spécification, il y a eu une certaine confusion sur si elle serait utile étant donné que, par exemple, les types individuels de supports MIME utilisés dans la messagerie et avec HTTP ont leurs propres règles sur les types de caractères UTF-8 et leur normalisation, et que les protocoles de transport d'application imposent leurs propres conventions sur les terminaisons de ligne. Il y a trois réponses. La première est que, rétrospectivement, il aurait été préférable d'avoir ces protocoles et leur type de contenu normalisés de la façon spécifiée ici, bien qu'il soit certainement trop tard pour les changer maintenant. La seconde est qu'on a plusieurs protocoles qui dépendent soit de la conception originale de Telnet, soit d'autres arrangements exigeant une définition standard, interopérable de chaînes sans étiquettes de contenu spécifiques d'une sorte ou l'autre. Whois [RFC3912] est un exemple de ce groupe. Comme il est envisagé de les mettre à niveau pour les utilisations non ASCII, la présente spécification donne une référence normative qui assure la même stabilité que fournit NVT aux formes ASCII. La présente spécification est destinée à être utilisée par d'autres spécifications qui n'ont pas encore défini comment utiliser Unicode. Avoir une définition préférée standard de l'Internet pour les flux de texte Unicode – plutôt que juste une pour les codages de transmission -- peut aider à améliorer la

spécification et l'interopérabilité des protocoles à développer à l'avenir. La présente spécification n'est pas destinée à être utilisée avec des spécifications qui permettent déjà l'utilisation de UTF-8 et définissent précisément cette utilisation.

5.2 Versions Unicode et applicabilité

L'IETF est en face d'un dilemme pratique à l'égard des versions de Unicode. Chaque nouvelle version apporte avec elle de nouveaux caractères et parfois de nouvelles combinaisons de caractères. La version 5.0 introduit le nouveau concept de séquences de caractères désignés comme si ils étaient des caractères individuels (voir [NamedSequences]). La normalisation représentée par NFC est stable si toutes les chaînes sont transmises et mémorisées en forme normalisée si des corrections ne sont jamais faites aux définitions de caractères ou aux tableaux de normalisation et si les codets non alloués ne sont jamais utilisés. Cette dernière condition est importante parce que un codet non alloué se normalise toujours comme lui-même. Cependant, si le même codet est alloué à un caractère dans une future version, il peut participer à une autre transposition de normalisation (certaines difficultés spécifiques à cet égard sont discutées dans la [RFC4690]). On notera que la transmission en forme normalisée n'est pas exigée dans la norme UTF-8 de l'IETF [RFC3629] ni par les normes qui dépendent de la version actuelle de Stringprep [RFC3454].

Tout irait bien avec ce qui est décrit à la Section 4 sauf pour un problème : les applications ne font normalement pas leurs propres conversions en Unicode et peuvent ne pas effectuer leurs propres normalisations mais plutôt s'appuyer sur leur système d'exploitation ou leur fonctions de bibliothèque de langues -- fonctions qui peuvent être mise à niveau ou autrement changées sans changement au code d'application lui-même. Par conséquent, il peut n'y avoir pas de moyen plausible pour qu'une application sache quelle version d'Unicode, ou quelle version des procédures de normalisation, elle utilise, ni de moyen par lequel elle puisse garantir que les deux vont être cohérentes.

À cause des changements des définitions et des tableaux par version, Stringprep et les documents qui en dépendent sont maintenant liés à Unicode version 3.2 [Unicode32] et la pleine interopérabilité de l'UTF-8 Internet standard [RFC3629], quand utilisé avec la normalisation spécifiée ici, dépend des définitions de normalisation et de ce que la définition de UTF-8 lui-même ne change pas après Unicode version 5.0. Ces hypothèses semble très sûres, mais ce sont quand même des hypothèses. Plutôt que d'être lié à la dernière version disponible de Unicode, version 5.0 [Unicode] ou de concepts plus large d'indépendance à la version sur la base d'hypothèses et conditions spécifiques, la présente spécification pourrait raisonnablement avoir été liée, comme Stringprep et Nameprep à Unicode 3.2 [Unicode32] ou une version intermédiaire plus récente, mais, en plus des inconvénient évidents d'avoir des normes différentes de l'IETF liées à des versions différentes de Unicode, le comportement de mise en œuvre d'application fondée sur la bibliothèque décrit ci-dessus rend ces liens à la version presque insignifiants en pratique.

En théorie, on peut contourner ce problème de quatre façons :

1. Se fixer sur une version particulière d'Unicode et essayer d'insister pour que les applications appliquent cette version, par exemple, en contenant des listes de caractères non alloués et en interdisant leur utilisation. Bien sûr, cela interdirait toute évolution pour inclure l'ajout de nouveaux scripts et les tableaux de codets non alloués seraient encombrants.

2. Exiger que chaque chaîne de "texte" ou fichier Unicode commence par une indication de version, un peu comme l'indicateur de "marque d'ordre des octets". Il est peu probable que cette disposition soit praticable. Plus important, elle exigerait que chaque mise en œuvre d'application soit prête à prendre en charge plusieurs tableaux et versions de normalisation ou qu'elle rejette le texte provenant de versions Unicode qu'elle n'est pas prête à traiter.

3. Concevoir un ensemble différent de règles de normalisation qui, par exemple, garantirait qu'aucun caractère alloué à un codet précédemment non alloué dans Unicode ne sera jamais normalisé en autre choix que lui-même et utiliser ces règles au lieu de NFC. Il n'est pas clair qu'un tel ensemble de règles soit possible ou si un autre ensemble de règles complètement stable pourrait être envisagé, peut-être en combinaison avec des restrictions sur la façon dont les caractères seraient ajoutés dans les futures versions de Unicode.

4. Envisager un processus de normalisation qui soit par ailleurs équivalent de NFC mais qui rejette les codets qui sont non alloués dans la version actuelle d'Unicode, plutôt que de transposer ces codets en eux-mêmes. Cela laisserait quand même un risque de corrections incompatibles dans Unicode et éventuellement quelques cas particuliers, mais il est probablement assez stable pour l'usage de l'Internet dans l'immense majorité des cas. Ce processus a été discuté dans le Consortium Unicode sous le nom de "NFC stable".

Aucune de ces approches ne semble idéale : la procédure idéale devrait être aussi stable et prévisible que l'a été l'ASCII. Mais ce niveau n'est simplement pas faisable tant que Unicode continue d'évoluer par l'ajout de nouveaux codets et scripts. La quatrième option mentionnée ci-dessus paraît être un compromis raisonnable.

6. Considérations sur la sécurité

La présente spécification fournit une forme standard pour l'utilisation de Unicode comme "texte du réseau". La plupart des mêmes questions de sécurité qui s'appliquent à UTF-8, discutées dans la [RFC3629], s'y appliquent, bien qu'elle devrait être légèrement moins sujette à certains risques du fait qu'elle exige la normalisation NFC et est généralement un peu plus restrictive. Cependant, les glissements de versions Unicode, discutés au paragraphe 5.2, peuvent introduire d'autres problèmes de sécurité.

Les programmes qui reçoivent ces flux devraient exercer une extrême prudence si elle supposent que des données entrantes sont normalisées, car il serait possible d'utiliser des formes non normalisées, ainsi que de l'UTF-8 invalide, au titre d'une attaque. En particulier, les pare-feu et autres systèmes qui interprètent les flux UTF-8 devraient être développés avec la connaissance claire qu'un attaquant peut délibérément envoyer du texte non normalisé, par exemple, pour éviter la détection par des systèmes de confrontation de texte sans malice.

NVT contient une exigence, dont la nécessité est répétée ici (voir la Section 2) que le caractère CR doit être immédiatement suivi d'un LF ou d'un ASCII NUL (un octet avec tous les bits à zéro). NUL peut être problématique pour certains langages de programmation qui l'utilisent comme terminaison de chaîne, et donc un piège pour les insouciants, sauf si la prudence est utilisée. Cela peut être une raison supplémentaire d'éviter complètement l'utilisation de CR, sauf en séquence avec LF, comme suggéré ci-dessus.

La discussion sur les versions Unicode (voir la Section 4 et le paragraphe 5.2) fait plusieurs hypothèses sur les futures versions d'Unicode, sur l'application appropriée de la normalisation NFC, et sur le traitement et la transmission de UTF-8 exactement comme spécifié dans la RFC 3629. Si une de ces hypothèses n'est pas correcte, il y a alors des cas dans lesqeuls de chaînes qui auraient été considérées équivalente ne se comparent pas comme égales. Un code robuste devrait être prêt à ces possibilités.

7. Remerciements

Tous nos remerciement à Mark Davis, Martin Duerst, et Michel Suignard pour leurs suggestions sur la normalisation de Unicode qui a conduit au format décrit ici, et en particulier à Mark qui a fourni les paragraphes décrivant le rôle de NFC. Merci aussi à Mark, Doug Ewell, Asmus Freytag pour avoir corrigé le texte qui décrit les formes de transmission de Unicode, et à Tim Bray, Carsten Bormann, Stephane Bortzmeyer, Martin Duerst, Frank Ellermann, Clive D.W. Feather, Ted Hardie, Bjoern Hoehrmann, Alfred Hoenes, Kent Karlsson, Bill McQuillan, George Michaelson, Chris Newman, et Marcos Sanz pour un certain nombre de commentaires utiles et de demandes d'éclaircissements.

Appendice A. Historique et contexte

Cet appendice contient une revue des travaux antérieurs de l'ARPANET et de l'Internet pour établir un type de texte standard, travail qui établit le contexte et la motivation de l'approche suivie dans ce document. Le texte est explicatif plutôt que normatif : rien dans cette section n'est destiné à changer ou mettre à jour une spécification actuelle. Ceux qui ne sont pas intéressés par cette revue et analyse peuvent en toute sécurité sauter cette section.

Une des premières décisions de conception d'application prise dans le développement de l'ARPANET, décision qui a été importée dans l'Internet, était la décision de normaliser sur un seul et très spécifique codage pour que le "texte" soit passé à travers le réseau [RFC0020]. Les hôtes du réseau étaient alors responsables de traduire ou transposer à partir de toute convention de codage de caractère utilisée localement en cette représentation intermédiaire commune, les hôtes envoyeurs la transposant et les hôtes receveurs transposant d'elle en leur forme locale comme nécessaire . Il est intéressant de noter qu'au moment du développement de l'ARPANET, les systèmes d'exploitation des hôtes participants utilisaient au moins trois normes de codage de caractères différentes : l'antique décimal codé binaire (BCD, *Binary Coded Decimal*) le alors dominant code d'échange BCD étendu (EBCDIC, *Extended BCD Interchange Code*) soutenu par les fabricants majeurs, et le code standard américain pour les échanges d'information (ASCII, *American Standard Code for Information Interchange*) alors encore émergeant. Comme ARPANET était un projet "ouvert" et que EBCDIC était intimement lié à un fabricant de matériel particulier, le groupe de travail réseau original s'est mis d'accord pour que la norme devrait être l'ASCII. Cette forme d'ASCII était précisément l'ASCII à 7 bits dans un champ de 8 bits, qui était en effet un compromis

entre les hôtes qui étaient en 7 bits natif (par exemple, avec cinq caractères de sept bits dans un mot de 36 bits) ceux qui étaient en 8 bits (utilisant des caractères de huit bits) et ceux qui avaient placé les caractères ASCII de 7 bits dans des champs de 9 bits avec deux bits de zéro en tête (quatre caractères dans un mot de 36 bits).

Plus de normalisation a été suggérée dans la première description préliminaire du protocole Telnet [RFC0097]. Avec les itérations de ce protocole [RFC0137] [RFC0139] et le dessin d'une définition essentiellement formelle un peu plus tard [RFC0318], une abstraction standard, le terminal virtuel de réseau (NVT, *Network Virtual Terminal*) a été établie. Les conventions de codage de caractère (initialement appelé "ASCII Telnet" et plus tard appelé "ASCII NVT", ou, plus précisément , "ASCII du réseau") incluaient l'exigence que le retour-chariot suivi par le saut à la ligne (CRLF) soit la représentation commune pour les fins de lignes de texte (étant donné que certains systèmes d'exploitation des "hôtes" participants utilisaient l'un de façon native, d'autres l'autre, et au moins un utilisait les deux, et quelques uns ne les utilisaient ni l'un ni l'autre (préférant à la place des lignes de longueur variable avec des délimiteurs de comptes ou des délimiteurs spéciaux ou des marqueurs) et spécifiaient des conventions pour certains autres caractères. Aussi, comme l'ASCII NVT était restreint à des caractères de sept bits, l'utilisation du bit de poids fort dans les octets était réservée pour la transmission des informations de signalisation de contrôle.

À très haut niveau, le concept était qu'un système pouvait utiliser tout codage de caractères de représentations de ligne approprié en local, mais le texte transmis sur le réseau comme du texte doit se conformer à la seule convention du "terminal virtuel de réseau". Virtuellement tous les premiers protocoles Internet qui pensaient transférer du "texte" acceptaient ce modèle du terminal virtuel, bien que chacun l'accepte ou le limite de différentes façons. Telnet, le flux de commandes et le type ASCII dans FTP [RFC0542], le flux de messages dans le transfert SMTP [RFC2821], et les chaînes passées à "finger" [RFC0742] et "whois" [RFC0954] sont les exemples classiques. Plus récemment, HTTP [RFC1945] [RFC2616] a suivi le même modèle général mais permet des données de 8 bits et laisse la séquence de fin de ligne non spécifiée (ce qui a été la source d'un nombre significatif de problèmes).

Appendice B. Définition de l'ASCII NVT

Le corps principal de la présente spécification est destiné à mettre à jour, et à donner une version internationalisée de la définition de l'ASCII du réseau. La spécification est auto-contenue dans les parties de la définition du Net-ASCII qui ne sont plus recommandées et ne sont pas incluses ci-dessus. Parce que Net-ASCII a un peu évolué au cours du temps et qu'il y a eu des débats sur quelle spécification est le Net-ASCII "officiel", il est approprié de revoir ici les éléments clés de cette définition. Cette revue est informelle à l'égard du contenu du Net-ASCII et ne devrait pas être considérée comme une mise à jour normative ou un résumé des spécifications antérieures (la Section 2 spécifie certaines mises à jour normatives de ces spécifications et les commentaires ci-dessous sont cohérents avec elle).

La première partie de la section intitulée "Imprimante et clavier NVT" de la [RFC0854] est généralement, bien que pas universellement, considérée comme étant la définition normative du terminal virtuel de réseau (ASCII) et donc du Net-ASCII. Elle inclut non seulement les caractères graphiques ASCII mais aussi un certain nombre de caractères de contrôle. Ces derniers ont une signification spécifique de l'Internet qui est souvent plus spécifique que les définitions de la spécification ASCII. Dans l'usage d'aujourd'hui, et pour la présente spécification, les précisions et mises à jour suivantes de cette liste devraient être notées. Chacune est accompagnée d'une brève explication de la raison pour laquelle la spécification originale n'est plus appropriée.

1. Les codes "définis mais pas exigés" -- BEL (U+0007), BS (U+0008), HT (U+0009), VT (U+000B), et FF (U+000C) -- et les codes de contrôle indéfinis ("C0") NE DEVRAIENT PAS être utilisés sauf exigé par des circonstances exceptionnelles. Soit leur définition originale "d'imprimante du réseau" ne sont plus d'utilisation générale, la pratique courante ayant abandonné les formats qui y sont spécifiés, soit que leur utilisation pour simuler des caractères qui sont mieux traités par Unicode n'est plus appropriée. Bien que l'apparition de certains de ces caractères dans la liste puisse sembler surprenante, BS a maintenant une interprétation ambiguë en pratique (écrasement dans certains systèmes mais pas dans d'autres) la largeur associée à HT varie selon l'environnement, et VT et FF n'ont pas un effet uniforme à l'égard du positionnement vertical ou du résultat de position horizontale associée. Bien sûr, les échappements telnet ne sont pas considérés comme faisant partie du flux de données et donc ne sont pas affectés par cette disposition.

2. Dans le Net-ASCII, CR NE DOIT PAS apparaître sauf immédiatement suivi par un NUL ou LF, ce dernier (CR LF) désignant la fonction de "nouvelle ligne". Aujourd'hui et comme spécifié ci-dessus, CR devrait généralement seulement apparaître quand il est suivi de LF. Parce que la disposition de page est mieux faite d'autres façons, parce que NUL a une interprétation spéciale dans certains langages de programmation, et pour éviter d'autres types de confusion, CR NUL devrait de préférence être évité comme spécifié ci-dessus.

3. LF CR NE DEVRAIT PAS apparaître sauf comme effet collatéral de plusieurs séquences de CR LF (par exemple, CR LF CR LF).

4. Les documents NVT historiques ne précisent pas de traitement particulier pour "LF nu" (LF sans CR) ou HT. Tous deux ont été généralement compris comme étant problématiques. Dans le cas de LF, il y a une différence d'interprétation sur si sa sémantique implique "aller à la même position sur la ligne suivante" ou "aller à la première position de la ligne suivante" et des considérations d'interopérabilité suggèrent de ne pas dépendre de l'interprétation que le receveur applique. En même temps, une mauvaise interprétation du LF est moins dommageable qu'une mauvaise interprétation du CR "nu" : dans le cas du CR, du texte peut être écrasé ou rendu complètement illisible ; dans celui du LF, la pire conséquence est un affichage d'une allure très bizarre. Évidemment, HT est problématique parce que il n'y a pas de façon standard de transmettre la position de tabulation prévue ou les informations de largeur dans le texte courant. Là encore, le dommage a peu de chance d'être grand si HT est simplement interprété comme une ou plusieurs espaces, mais, en général, on ne peut pas s'appuyer dessus pour des informations de format.

On notera que le caractère telnet IAC (un octet ne comportant que des uns, c'est-à-dire, %xFF) n'est lui-même pas un problème pour UTF-8 car cet octet particulier ne peut pas apparaître dans une chaîne UTF-8 valide. Cependant, bien que peu d'entre eux aient été utilisés, telnet permet que d'autres caractères introducteurs de commandes dont les séquences de bits dans un octet puissent faire partie de caractères UTF-8 valides. Bien que cela ne cause pas d'ambiguïté dans UTF-8, Unicode alloue un caractère graphique ("Latin Small Letter Y with Diaeresis") à U+00FF (octets C3 B0 en UTF-8). Une certaine prudence est clairement de mise dans ce domaine.

Appendice C. Le problème de la terminaison de ligne

La définition de comment une terminaison de ligne devrait être notée dans les chaînes de texte sur le réseau pour l'Internet a fait l'objet de controverses depuis bien avant l'introduction du NVT. Certains ont avancé que les receveurs devraient être obligés d'interpréter presque tout ce qu'un envoyeur pourrait vouloir comme terminaison de ligne comme une réelle terminaison de ligne. D'autres ont souligné que cela conduirait à des ambiguïtés d'interprétation et de présentation et violerait le principe qu'on devrait minimiser le nombre de formes permises sur le réseau afin de promouvoir l'interopérabilité et éliminer le problème de "chaque receveur doit comprendre tout format de l'envoyeur". La conception de la présente spécification, comme celle de NVT, adopte cette dernière approche. Ses concepteurs estiment qu'il y a peu de sens à un standard si il doit spécifier que "chacun peut faire ce qu'il veut et le receveur doit juste se débrouiller".

Plus de discussion sur la nature et l'évolution du problème de la terminaison de ligne apparaît au paragraphe 5.8 de la norme Unicode [Unicode] et il est suggéré de la lire. Si on commençait l'Internet aujourd'hui, il serait probablement raisonnable de suivre la recommandation d'utiliser LS (U+2028) exclusivement, de préférence à CRLF. Cependant, la base installée d'utilisations de CRLF et l'importance de la rétro compatibilité avec NVT et les protocoles qui l'utilisent rendent cela impossible, de sorte qu'il est nécessaire de continuer d'utiliser le CRLF comme "Fonction de nouvelle ligne" ("NLF", voir la section de terminologie dans cette référence).

Appendice D. Note sur les futurs travaux en relation

On devrait tenir compte d'une option Telnet (ou SSH [RFC4251]) pour spécifier ce type de flux et d'une extension à FTP [RFC0959] pour permettre un nouveau type de données "Unicode text".

Références

Références normatives

[ISO-10646] Norme ISO/CEI 10646, "Technologie de l'information – Jeu de caractères universel codé sur plusieurs octets (UCS) - Partie 1 : Architecture et plan multilingue de base", mai 1993.

[NFC] Davis, M. and M. Duerst, "Unicode Standard Annex #15: Unicode Normalization Forms", octobre 2006, <http://www.unicode.org/reports/tr15/>.

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

(MàJ par RFC8174)

[RFC3629] F. Yergeau, "UTF-8, un format de transformation de la norme ISO 10646", STD 63, novembre 2003.

[RFC5234] D. Crocker, P. Overell, "BNF augmenté pour les spécifications de syntaxe : ABNF", janvier 2008. (STD0068)

[Unicode] The Unicode Consortium, "The Unicode Standard, Version 5.0", 2007. Boston, MA, USA: Addison-Wesley. ISBN 0-321-48091-0

[Unicode32] The Unicode Consortium, "The Unicode Standard, Version 3.0", 2000. (Reading, MA, Addison-Wesley, 2000. ISBN 0-201-61633-5). La version 3.2 consiste en la définition de ce livre tel qu'amendé par Unicode Standard Annex n° 27: Unicode 3.1 (http://www.unicode.org/reports/tr27/) et par Unicode Standard Annex n° 28: Unicode 3.2 (http://www.unicode.org/reports/tr28/).

Références pour information

[ASCII] American National Standards Institute (anciennement United States of America Standards Institute), "USA Code for Information Interchange", ANSI X3.4-1968, 1968. ANSI X3.4-1968 a été remplacé par de nouvelles versions avec des modifications, mais la version 1968 reste d'autorité pour l'Internet. La norme ISO 646 [ISO.646.1991] est généralement considérée comme équivalente à ASCII.

[ISO.646.1991] Organisation Internationale de Normalisation, "Technologie de l'information - Jeu de caractères ISO codé sur 7 bits pour les échanges d'informations", Norme ISO 646, 1991.

[NamedSequences] The Unicode Consortium, "NamedSequences-4.1.0.txt", 2005, <http://www.unicode.org/Public/UNIDATA/NamedSequences.txt>.

[RFC0020] V. Cerf, "Format ASCII pour les échanges sur les réseaux", octobre 1969. (STD80)

[RFC0097] J. Melvin et R. Watson, "Premières réactions à la proposition de protocole Telnet", février 1971.

[RFC0137] T. O'Sullivan, "Document de proposition du protocole Telnet", avril 1971.

RFC0139] T. O'Sullivan, "Discussion du protocole Telnet", avril 1971.

[RFC0318] J. Postel, "Protocoles Telnet", avril 1972.

[RFC0542] N. Neigus, "Protocole de transfert de fichiers", août 1973.

[RFC0698] T. Mock, "Option Telnet ASCII étendu", juillet 1975. *(Rendue obsolète par la* RFC5198*)*

[RFC0742] K. Harrenstien, "NAME/FINGER", décembre 1977. *(Obsolète, voir la* RFC1288*)*

[RFC0854] J. Postel et J. Reynolds," Spécification du protocole TELNET", STD 8, mai 1983.

[RFC0954] K. Harrenstien, M. Stahl, E. Feinler, "NICNAME/Qui-est-qui", octobre 1985. *(Rendue obsolète par* 3912)

[RFC0959] J. Postel et J. Reynolds, "Protocole de transfert de fichiers (FTP)", STD 9, octobre 1985. *(MàJ par* RFC7151)

[RFC1945] T. Berners-Lee, R. Fielding, H. Frystyk, "Protocole de transfert Hypertext -- HTTP/1.0", mai 1996. *(Info.)*

[RFC2277] H. Alvestrand, "Politique de l'IETF en matière de jeux de caractères et de langages", BCP 18, janvier 1998.

[RFC2616] R. Fielding et autres, "Protocole de transfert hypertexte -- HTTP/1.1", juin 1999. *(D.S., MàJ par* 2817, 6585)

[RFC2781] P. Hoffman et F. Yergeau, "UTF-16, un codage de la norme ISO 10646", février 2000.

[RFC2821] J. Klensin, éditeur, "Protocole simple de transfert de messagerie", STD 10, avril 2001. *(Obsolète, voir*

RFC5321)

[RFC3454] P. Hoffman et M. Blanchet, "Préparation de chaînes internationalisées ("stringprep")", décembre 2002. *(P.S.)*

[RFC3491] P. Hoffman et M. Blanchet, "Nameprep : Profil Stringprep pour les noms de domaine internationalisés (IDN)", mars 2003. (*Remplacée par la* RFC5891, *P.S.*)

[RFC3912] L. Daigle, "Spécification du protocole WHOIS", septembre 2004. *(D.S.)*

[RFC4251] T. Ylonen et C. Lonvick, "Architecture du protocole Secure Shell (SSH)", janvier 2006. *(P.S. ; MàJ par* RFC8308)

[RFC4690] J. Klensin et autres, "Révisons et recommandations pour les noms de domaines internationalisés (IDN)", septembre 2006. *(Information)*

Adresse des auteurs

John C Klensin
1770 Massachusetts Ave, #322
Cambridge, MA 02140
USA
téléphone : +1 617 491 5735
mél : john-ietf@jck.com

Michael A. Padlipsky
8011 Stewart Ave.
Los Angeles, CA 90045
USA
téléphone : +1 310-670-4288
mailto:the.map@alum.mit.edu

Déclaration complète de droits de reproduction