

RFC 5234 : Augmented BNF for Syntax Specifications: ABNF

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 Février 2008

Date de publication du RFC : Janvier 2008

<http://www.bortzmeyer.org/5234.html>

Ce RFC fait partie des RFC ancillaires, qui ne spécifient pas directement un protocole IETF mais fournissent des outils pour les "vrais" RFC. En l'occurrence, il normalise le mini-langage pour écrire des grammaires.

Beaucoup de RFC doivent spécifier un langage, en général assez simple (jamais de la taille d'un langage de programmation) mais néanmoins suffisamment important pour que les descriptions informelles du langage soient risquées. Depuis longtemps, on utilise en informatique des notations dérivées du langage BNF pour spécifier formellement un langage. Le problème est qu'il existe plusieurs dialectes de BNF (comme EBNF) et que les RFC ont besoin d'une référence stable. À une époque, chaque RFC définissait approximativement sa grammaire formelle et l'utilisait ensuite. ABNF est issue d'un effort de Dave Crocker de regroupement de la définition de la grammaire en un seul RFC, le RFC 2234¹ en 1997, devenu ensuite le RFC 4234 en 2006. Notre RFC succède au fameux RFC 4234 (avec très peu de changements, le principal étant le passage au statut de norme IETF à part entière, "*Full Standard*") et qui décrit **ABNF**, le dialecte IETF de BNF. Classique sur beaucoup de points, ce dialecte a quand même quelques variations, issues d'une histoire très ancienne. Par exemple, le signe — pour le choix est remplacé par /.

Quelques outils sont disponibles <<http://tools.ietf.org/inventory/verif-tools>> pour aider les auteurs de grammaires. Mais je trouve que c'est encore insuffisant. S'il existe deux vérificateurs (qui peuvent tester qu'une grammaire est cohérente), il n'existe guère de générateurs d'analyseurs syntaxiques. En revanche, à des fins de test <<http://www.bortzmeyer.org/test-logiciel.html>>, il existe au moins deux programmes, Eustathius <<http://www.bortzmeyer.org/eustathius-test-grammars.html>> et abnfgn <<http://www.quut.com/abnfgn/>>, qui génèrent automatiquement des exemples

1. Pour voir le RFC de numéro NNN, <http://www.ietf.org/rfc/rfcNNN.txt>, par exemple <http://www.ietf.org/rfc/rfc2234.txt>

à partir d'une grammaire. Vous trouverez de nombreux exemples de grammaires ABNF dans les sources d'Eustathius.

Notre RFC normalise non seulement le format des grammaires mais aussi une bibliothèque de productions prêtes à l'emploi comme HEXDIG pour les caractères utilisables pour écrire de l'hexadécimal ou ALPHA pour les lettres de l'alphabet latin. Baptisée "*Core*", cette bibliothèque est décrite dans l'appendice B.

À titre d'exemple, voici la spécification de SPF (décrit dans le RFC 4408) en ABNF :

```

record          = version terms *SP
version         = "v=spf1"

terms           = *( 1*SP ( directive / modifier ) )

directive       = [ qualifieur ] mechanism
qualifieur     = "+" / "-" / "?" / "~"
mechanism       = ( all / include
                  / A / MX / PTR / IP4 / IP6 / exists )

all             = "all"
include        = "include" ":" domain-spec
A              = "a"      [ ":" domain-spec ] [ dual-cidr-length ]
MX             = "mx"     [ ":" domain-spec ] [ dual-cidr-length ]
PTR            = "ptr"    [ ":" domain-spec ]
IP4            = "ip4"    ":" ip4-network  [ ip4-cidr-length ]
IP6            = "ip6"    ":" ip6-network  [ ip6-cidr-length ]
exists         = "exists" ":" domain-spec

modifier        = redirect / explanation / unknown-modifier
redirect        = "redirect" "=" domain-spec
explanation      = "exp" "=" domain-spec
unknown-modifier = name "=" macro-string

ip4-cidr-length = "/" 1*DIGIT
ip6-cidr-length = "/" 1*DIGIT
dual-cidr-length = [ ip4-cidr-length ] [ "/" ip6-cidr-length ]

ip4-network     = qnum "." qnum "." qnum "." qnum
qnum            = DIGIT ; 0-9
                / %x31-39 DIGIT ; 10-99
                / "1" 2DIGIT ; 100-199
                / "2" %x30-34 DIGIT ; 200-249
                / "25" %x30-35 ; 250-255
                ; conventional dotted quad notation. e.g., 192.0.2.0
ip6-network     = <as per [RFC 3513], section 2.2>
                ; e.g., 2001:DB8::CD30

domain-spec     = macro-string domain-end
domain-end      = ( "." toplabel [ "." ] ) / macro-expand
toplabel        = ( *alphanumeric ALPHA *alphanumeric ) /
                ( 1*alphanumeric "-" *( alphanumeric / "-" ) alphanumeric )
                ; LDH rule plus additional TLD restrictions
                ; (see [RFC3696], Section 2)

alphanumeric    = ALPHA / DIGIT

explain-string  = *( macro-string / SP )

macro-string    = *( macro-expand / macro-literal )
macro-expand    = ( "%{" macro-letter transformers *delimiter "}" )
                / "%%" / "%_" / "%-"
macro-literal    = %x21-24 / %x26-7E

```

```

macro-letter      = "s" / "l" / "o" / "d" / "i" / "p" / "h" /
                  "c" / "r" / "t"
transformers      = *DIGIT [ "r" ]
delimiter        = "." / "-" / "+" / "," / "/" / "_" / "="

name             = ALPHA *( ALPHA / DIGIT / "-" / "_" / "." )

header-field     = "Received-SPF:" [CFWS] result FWS [comment FWS]
                  [ key-value-list ] CRLF

result           = "Pass" / "Fail" / "SoftFail" / "Neutral" /
                  "None" / "TempError" / "PermError"

key-value-list   = key-value-pair *( ";" [CFWS] key-value-pair )
                  [ ";" ]

key-value-pair   = key [CFWS] "=" ( dot-atom / quoted-string )

key              = "client-ip" / "envelope-from" / "helo" /
                  "problem" / "receiver" / "identity" /
                  mechanism / "x-" name / name

identity         = "mailfrom"      ; for the "MAIL FROM" identity
                  / "helo"         ; for the "HELO" identity
                  / name           ; other identities

dot-atom        = <unquoted word as per [RFC2822]>
quoted-string   = <quoted string as per [RFC2822]>
comment         = <comment string as per [RFC2822]>
CFWS            = <comment or folding white space as per [RFC2822]>
FWS             = <folding white space as per [RFC2822]>
CRLF            = <standard end-of-line token as per [RFC2822]>

```

On notera qu'ABNF ne spécifie pas la représentation sous forme de bits, mais une structure abstraite. Ce point, qui n'était pas clair dans les premières versions de la norme, et qui est souvent oublié par les utilisateurs d'ABNF, est décrit dans la section 2.4. ABNF spécifie des caractères, pas des octets, et la production :

```
comment-start = %3b
```

décrit bien le caractère point-virgule, pas forcément qu'il sera représenté par un octet de valeur numérique 3B (59 en décimal). Un encodage d'Unicode comme UTF-32, par exemple, le codera différemment

L'un des rares changements par rapport à son prédécesseur, le RFC 4234, aura été l'ajout d'un avertissement à la production LWSP ("*Linear White Space*", qui permet des espaces en début de ligne). Cette production, décrite dans l'appendice B, qui définit la bibliothèque standard de productions, a en effet été critiquée comme trop puissante : des auteurs de RFC l'ont utilisée sans comprendre ses conséquences. La définition comprend désormais l'avertissement suivant, qui a été négocié à la virgule près, avec acharnement : "*Use of this linear-white-space rule permits lines containing only white space that are no longer legal in mail headers and have caused interoperability problems in other contexts. Do not use when defining mail headers and use with caution in other contexts.*"

Notons qu'il n'y a pas grand'chose d'obligatoire à l'IETF. Aucune autorité ne peut dire aux auteurs de RFC « Désormais, vous êtes obligés d'utiliser ABNF ». Ceux qui le font le choi[Caractère Unicode

non montré²]sissent volontairement. La principale « dissidence » vient du RFC 2616, qui normalise HTTP, et qui utilise un langage différent.

Notons aussi que, bien que l'un des principaux avantages des langages formels soit la possibi[Caractère Unicode non montré]lité de vérification automatique, cette possibi[Caractère Unicode non montré]lité n'est pas utilisée systématiquement. C'est ainsi que certains RFC ont été publiés avec des bogues dans leur grammaire ABNF...

Le RFC 2234 était « Proposition de norme », le RFC 4234 était « Projet de norme ». Pour franchir l'étape supplémentaire menant au statut de « Norme » tout court, ABNF a dû subir tout un processus, mal[Caractère Unicode non montré]gré l'extrême ancienneté de cette norme. Notamment, ABNF a dû subir l'examen des programmes qui le mettent en œuvre, examen décrit en <http://ietf.org/IESG/Implementations/RFC4234_implem.txt>.

2. Car trop difficile à faire afficher par L^AT_EX