Network Working Group
Internet-Draft
Obsoletes: 2425, 2426, 4770
(if approved)
Intended status: Standards Track
Expires: August 7, 2008

P. Resnick
QUALCOMM Incorporated
S. Perreault
Viagenie

February 4, 2008

vCard Format Specification
draft-resnick-vcarddav-vcardrev-01

Status of This Memo

Copyright Notice

Abstract

This document defines the vCard data format for representing and exchanging a variety of information about an individual (e.g., formatted and structured name and delivery addresses, email address, multiple telephone numbers, photograph, logo, audio clips, etc.).

Table of Contents

1. Introduction

Note: This draft contains much of the same text as 2425 and 2426 which may not be correct. Those two RFCs have been merged and the structure of this draft is what's new. Some vCard-specific suggestions have been added, but for the most part this is still very open. But we'd like to get feedback on the structure mostly so that it may be fixed.

Electronic address books have become ubiquitous. Their increased presense on portable, connected devices as well as the diversity of platforms exchanging contact data call for a standard. This memo defines the vCard format, which allows the capture and exchange of information normally stored within an address book or directory application.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. MIME Type Registration

To: ietf-types@iana.org

Subject: Registration of media type text/vcard

Type name: text

Subtype name: vcard

Required parameters: none

Optional parameters: charset

Encoding considerations: The "charset" MIME parameter is interpreted as defined in [RFC2046], section 4.1.2. If it is omitted, the default encoding is UTF-8 as defined in [RFC3629].

Security considerations: See Section 8.

Interoperability considerations: The text/vcard media type is intended to identify vCard data of any version. There are older specifications of vCard [RFC2426][oldreference_VCARD] still in common use. While these formats are similar, they are not strictly compatible. In general, it is necessary to inspect the value of the VERSION property (see Section 5.7.9) for identifying

the standard to which a given vCard object conforms.

In addition, the following media types are known to have been used to refer to vCard data. They should be considered deprecated in favor of text/vcard.

* text/directory

* text/directory; type=vcard

* text/x-vcard

Published specification: draft-resnick-vcarddav-vcardrev-01

Applications that use this media type: They are numerous, diverse, and include mail user agents, instant messaging clients, address book applications, directory servers, customer relationship management software, etc.

Additional information:

Magic number(s):

File extension(s): .vcf

Macintosh file type code(s):

Person & email address to contact for further information: Simon Perreault <simon.perreault@viagenie.ca>

Intended usage: COMMON

Restrictions on usage: none

Author: Pete Resnick and Simon Perreault

Change controller: IETF

4. vCard Format Specification

The text/vcard MIME content type (hereafter known as "vCard") contains contact information, typically pertaining to a single contact or group of contacts. The content consists of one or more lines in the format given below.

4.1. Line Delimiting and Folding

Individual lines within vCard are delimited by the [RFC2822] line break, which is a CRLF sequence (ASCII decimal 13, followed by ASCII decimal 10). Long logical lines of text can be split into a multiple-physical-line representation using the following folding technique. After generating a content line, lines longer than 75 characters SHOULD be folded.

A logical line MAY be continued on the next physical line anywhere between two characters by inserting a CRLF immediately followed by a single white space character (space, ASCII decimal 32, or horizontal tab, ASCII decimal 9). At least one character must be present on the folded line. Any sequence of CRLF followed immediately by a single white space character is ignored (removed) when processing the content type. For example the line:

```
DESCRIPTION:This is a long description that exists on a long line.
```

can be represented as:

```
DESCRIPTION:This is a long description
 that exists on a long line.
```

It could also be represented as:

```
DESCRIPTION:This is a long descrip
 tion that exists o
 n a long line.
```

The process of moving from this folded multiple-line representation of a property definition to its single line representation is called unfolding. Unfolding is accomplished by regarding CRLF immediately followed by a white space character (namely HTAB ASCII decimal 9 or SPACE ASCII decimal 32) as equivalent to no characters at all (i.e., the CRLF and single white space character are removed).

Folding is done after any content encoding of a type value. Unfolding is done before any decoding of a type value in a content line.

4.2. ABNF Format Definition

The following ABNF uses the notation of [RFC5234], which also defines CRLF, WSP, DQUOTE, VCHAR, ALPHA, and DIGIT. After the unfolding of any folded lines as described above, the syntax for a line of this content type is as follows:

```
   contentline  = name *(";" param) ":" value CRLF
      ; When parsing a content line, folded lines MUST first
      ; be unfolded according to the unfolding procedure
      ; described above.
      ; When generating a content line, lines longer than 75
      ; characters SHOULD be folded according to the folding
      ; procedure described above.

   name         = x-name / iana-token

   iana-token   = 1*(ALPHA / DIGIT / "-")
      ; identifier registered with IANA

   x-name       = "x-" 1*(ALPHA / DIGIT / "-")
      ; Names that begin with "x-" or "X-" are
      ; reserved for experimental use, not intended for released
      ; products, or for use in bilateral agreements.

   param        = param-name "=" param-value *("," param-value)

   param-name   = x-name / iana-token

   param-value  = ptext / quoted-string

   ptext  = *SAFE-CHAR

   value = *VALUE-CHAR
         / valuespec      ; valuespec defined in section 5.8.4

   contentline  = name *(";" param) ":" value CRLF
      ; When parsing a content line, folded lines MUST first
      ; be unfolded according to the unfolding procedure
      ; described above.
      ; When generating a content line, lines longer than 75
      ; characters SHOULD be folded according to the folding
      ; procedure described above.

   name         = x-name / iana-token

   iana-token   = 1*(ALPHA / DIGIT / "-")
      ; identifier registered with IANA

   x-name       = "x-" 1*(ALPHA / DIGIT / "-")
      ; Names that begin with "x-" or "X-" are
      ; reserved for experimental use, not intended for released
      ; products, or for use in bilateral agreements.

   param        = param-name "=" param-value *("," param-value)
```

```
param-name   = x-name / iana-token

param-value  = ptext / quoted-string

ptext  = *SAFE-CHAR

value = *VALUE-CHAR
      / valuespec      ; valuespec defined in section 5.8.4
```

A line that begins with a white space character is a continuation of the previous line, as described above. The white space character and immediately preceeding CRLF should be discarded when reconstructing the original line. Note that this line-folding convention differs from that found in [RFC2822], in that the sequence <CRLF><WSP> found anywhere in the content indicates a continued line and should be removed.

Property names and parameter names are case insensitive (e.g., the property name "fn" is the same as "FN" and "Fn"). Parameter values MAY be case sensitive or case insensitive, depending on their definition.

Each property defined in a vCard instance MAY have multiple values. The general rule for encoding multi-valued properties is to simply create a new content line for each value (including the property name). However, it should be noted that some value types support encoding multiple values in a single content line by separating the values with a comma ",". This approach has been taken for several of the content types defined below (date, time, integer, float), for space-saving reasons.

4.3. Value Types

Lists of values are delimited by a list delimiter, specified by the COMMA character (ASCII decimal 44). A COMMA character in a value MUST be escaped with a BACKSLASH character (ASCII decimal 92).

Compound type values are delimited by a field delimiter, specified by the SEMI-COLON character (ASCII decimal 59). A SEMI-COLON in a component of a compound property value MUST be escaped with a BACKSLASH character (ASCII decimal 92).

Standard value types are defined below.

```
  valuespec =  text-list
             / URI              ; from Appendix A of [RFC3986]
             / date-list
             / time-list
```

```
                / date-time-list
                / boolean
                / integer-list
                / float-list
                / binary
                / vcard
                / phone-number
                / utc-offset
                / iana-valuespec

     text-list = *TEXT-LIST-CHAR *("," *TEXT-LIST-CHAR)

     TEXT-LIST-CHAR = "\\" / "\," / "\n"
                   / <any VALUE-CHAR except , or \ or newline>
         ; Backslashes, newlines, and commas must be encoded.
         ; \n or \N can be used to encode a newline.

     date-list = date *("," date)

     time-list = time *("," time)

     date-time-list = date "T" time *("," date "T" time)

     boolean = "TRUE" / "FALSE"

     integer-list = integer *("," integer)

     integer = [sign] 1*DIGIT

     float-list = float *("," float)

     float = [sign] 1*DIGIT ["." 1*DIGIT]

     sign = "+" / "-"

     binary = <A "B" binary encoded string as defined by [RFC2047].>

     vcard = <vCard data encoded as specified below.>

     phone-number = <A telephone number as defined in
                    [CCITT.E163.1988] and [CCITT.X121.1988]>

     date = date-fullyear ["-"] date-month ["-"] date-mday

     date-fullyear = 4 DIGIT

     date-month = 2 DIGIT     ;01-12
```

```
date-mday = 2 DIGIT      ;01-28, 01-29, 01-30, 01-31
                         ;based on month/year

time = time-hour [":"] time-minute [":"] time-second [time-secfrac]
       [time-zone]

time-hour = 2 DIGIT      ;00-23

time-minute = 2 DIGIT    ;00-59

time-second = 2 DIGIT    ;00-60 (leap second)

time-secfrac = "," 1*DIGIT

time-zone = "Z" / time-numzone

time-numzome = sign time-hour [":"] time-minute

utc-offset = ("+" / "-") time-hour ":" time-minute

iana-valuespec = <a publicly-defined valuetype format, registered
                  with IANA, as defined in section 15 of this
                  document>
```

Some specific notes on the value types and formats:

"text": The "text" value type should be used to identify values that contain human-readable text. The character set in which the text is represented is controlled by the "charset" MIME type parameter. Note that there is no way to override this parameter on a per-property basis. As for the language, it is controlled by the "language" property parameter defined in Section 4.4.

Examples for "text":

```
this is a text value
this is one value,this is another
this is a single value\, with a comma encoded
```

A formatted text line break in a text value type MUST be represented as the character sequence backslash (ASCII decimal 92) followed by a Latin small letter n (ASCII decimal 110) or a Latin capital letter N (ASCII decimal 78), that is "\n" or "\N".

For example a multiple line DESCRIPTION value of:

```
Mythical Manager
Hyjinx Software Division
BabsCo, Inc.
```

could be represented as:

```
DESCRIPTION:Mythical Manager\nHyjinx Software Division\n
 BabsCo\, Inc.\n
```

demonstrating the \n literal formatted line break technique, the CRLF-followed-by-space line folding technique, and the backslash escape technique.

"uri": The "uri" value type should be used to identify values that are referenced by a URI (including a Content-ID URI), instead of encoded in-line. These value references might be used if the value is too large, or otherwise undesirable to include directly. The format for the URI is as defined in [RFC3986]. Note that the value of a property of type "uri" is what the URI points to, not the URI itself.

Examples for "uri":

```
http://www.foobar.com/my/picture.jpg
ldap://ldap.foobar.com/cn=babs%20jensen
```

"date", "time", and "date-time": Each of these value types is based on a subset of the definitions in [ISO.8601.1988] standard. Multiple "date" and "time" values can be specified using the comma-separated notation.

Examples for "date":

```
1985-04-12
1996-08-05,1996-11-11
19850412
```

Examples for "time":

```
10:22:00
102200
10:22:00.33
10:22:00.33Z
10:22:33,11:22:00
10:22:00-08:00
```

Examples for "date-time":

```
1996-10-22T14:00:00Z
1996-08-11T12:34:56Z
19960811T123456Z
1996-10-22T14:00:00Z,1996-08-11T12:34:56Z
```

"boolean": The "boolean" value type is used to express boolen values. These values are case insensitive.

Examples:

```
TRUE
false
True
```

"integer": The "integer" value type is used to express signed integers in decimal format. If sign is not specified, the value is assumed positive "+". Multiple "integer" values can be specified using the comma-separated notation.

Examples:

```
1234567890
-1234556790
+1234556790,432109876
```

"float": The "float" value type is used to express real numbers. If sign is not specified, the value is assumed positive "+". Multiple "float" values can be specified using the comma-separated notation.

Examples:

```
20.30
1000000.0000001
1.333,3.14
```

"binary": The "binary" value type specifies that the type value is inline, encoded binary data. This value type can be specified in the PHOTO, LOGO, SOUND, and KEY types.

If inline encoded binary data is specified, the ENCODING type parameter MUST be used to specify the encoding format. The binary data MUST be encoded using the "B" encoding format. Long lines of encoded binary data SHOULD BE folded to 75 characters using the folding method defined in Section 4.1.

"vcard": The "vcard" value type specifies that the type value is

another vCard. This value type can be specified in the AGENT property. The value type is defined by this specification. Since each of the type declarations within the vcard value type are being specified within a text value themselves, they MUST be terminated with the backslash escape sequence "\n" or "\N", instead of the normal newline character sequence CRLF. In addition, any COMMA character (ASCII decimal 44), SEMI-COLON character (ASCII decimal 59) and COLON character (ASCII decimal 58) MUST be escaped with the BACKSLASH character (ASCII decimal 92). For example, with the AGENT property a value would be specified as:

```
AGENT:BEGIN:VCARD\nFN:Joe Friday\nTEL:+1-919-555-7878\n
 TITLE:Area Administrator\, Assistant\n EMAIL\;TYPE=INTERN\n
 ET:jfriday@host.com\nEND:VCARD\n
```

"phone-number": The "phone-number" value type specifies that the type value is a telephone number. This value type can be specified in the TEL type. The value type is a text value that has the special semantics of a telephone number as defined in [CCITT.E163.1988] and [CCITT.X121.1988].

"utc-offset": The "utc-offset" value type specifies that the type value is a signed offset from UTC. This value type can be specified in the TZ type.

The value type is an offset from Coordinated Universal Time (UTC). It is specified as a positive or negative difference in units of hours and minutes (e.g., +hh:mm). The time is specified as a 24-hour clock. Hour values are from 00 to 23, and minute values are from 00 to 59. Hour and minutes are 2-digits with high order zeroes required to maintain digit count. The extended format for ISO 8601 UTC offsets MUST be used. The extended format makes use of a colon character as a separator of the hour and minute text fields.

4.4. Pre-defined Parameters

The following parameters are defined for general use.

```
predefined-param = encodingparm
                 / valuetypeparm
                 / languageparm

encodingparm = "encoding" "=" encodingtype

encodingtype = "b"       ; from [RFC2047]
             / iana-token  ; registered as described in
                           ; section 15 of this document

predefined-param = encodingparm
                 / valuetypeparm
                 / languageparm

encodingparm = "encoding" "=" encodingtype

encodingtype = "b"       ; from [RFC2047]
             / iana-token  ; registered as described in
                           ; section 15 of this document

predefined-param = encodingparm
                 / valuetypeparm
                 / languageparm

encodingparm = "encoding" "=" encodingtype

encodingtype = "b"       ; from [RFC2047]
             / iana-token  ; registered as described in
                           ; section 15 of this document
```

The "language" property parameter is used to identify data in multiple languages. There is no concept of "default" language, except as specified by any "Content-Language" MIME header parameter that is present. The value of the "language" property parameter is a language tag as defined in Section 2 of [RFC4646].

The "encoding" property parameter is used to specify an alternate encoding for a value. If the value contains a CRLF, it must be encoded, since CRLF is used to separate lines in the content-type itself. Currently, only the "b" encoding is supported.

The "b" encoding can also be useful for binary values that are mixed with other text information in the body part (e.g., a certificate). Using a per-value "b" encoding in this case leaves the other information in a more readable form. The encoded base 64 value can be split across multiple physical lines by using the line folding technique described above.

The Content-Transfer-Encoding header field is used to specify the encoding used for the body part as a whole. The "encoding" property parameter is used to specify an encoding for a particular value (e.g., a certificate). In this case, the Content-Transfer-Encoding header might specify "8bit", while the one certificate value might specify an encoding of "b" via an "encoding=b" property parameter.

The Content-Transfer-Encoding and the encodings of individual properties given by the "encoding" property parameter are independent of one another. When encoding a text/vcard body part for transmission, individual property encodings are performed first, then the entire body part is encoded according to the Content-Transfer-Encoding. When decoding a text/vcard body part, the Content-Transfer-Encoding is decoded first, and then any individual properties with an "encoding" property parameter are decoded.

The "value" parameter is optional, and is used to identify the value type (data type) and format of the value. The use of these predefined formats is encouraged even if the value parameter is not explicity used. By defining a standard set of value types and their formats, existing parsing and processing code can be leveraged. The predefined data type values MUST NOT be repeated in COMMA separated value lists except within the N, NICKNAME, ADR and CATEGORIES properties.

Including the value type explicitly as part of each property provides an extra hint to keep parsing simple and support more generalized applications. For example a search engine would not have to know the particular value types for all of the items for which it is searching. Because the value type is explicit in the definition, the search engine could look for dates in any item type and provide results that can still be interpreted.

5. vCard Properties

What follows is an enumeration of the standard vCard properties.

5.1. General Properties

5.1.1. BEGIN

Purpose: To denote the beginning of a syntactic entity within a text/vcard content-type.

Value type: text

Special notes: The content entity MUST begin with the BEGIN property with a value of "VCARD".

The BEGIN type is used in conjunction with the END type to delimit an entity containing a related set of properties within an text/vcard content-type. This construct can be used instead of or in addition to wrapping separate sets of information inside additional MIME headers. It is provided for applications that wish to define content that can contain multiple entities within the same text/vcard content-type or to define content that can be identifiable outside of a MIME environment.

Example:

```
BEGIN:VCARD
```

5.1.2. END

Purpose: To denote the end of a syntactic entity within a text/vcard content-type.

Value type: text

Special notes: The content entity MUST end with the END type with a value of "VCARD".

The END type is used in conjunction with the BEGIN type to delimit an entity containing a related set of properties within an text/vcard content-type. This construct can be used instead of or in addition to wrapping separate sets of information inside additional MIME headers. It is provided for applications that wish to define content that can contain multiple entities within the same text/vcard content-type or to define content that can be identifiable outside of a MIME environment.

Example:

```
END:VCARD
```

5.1.3. SOURCE

Purpose: To identify the source of directory information contained in the content type.

Value type: uri

Special notes: The SOURCE property is used to provide the means by which applications knowledgable in the given directory service protocol can obtain additional or more up-to-date information from the directory service. It contains a URI as defined in [RFC3986] and/or other information referencing the vCard to which the information pertains. When directory information is available from more than one source, the sending entity can pick what it considers to be the best source, or multiple SOURCE properties can be included.

Examples:

```
SOURCE:ldap://ldap.host/cn=Babs%20Jensen,%20o=Babsco,%20c=US

SOURCE:http://directory.example.com/addressbooks/jdoe/
 Jean%20Dupont.vcf
```

5.1.4. NAME

Purpose: To identify the displayable name of the directory entity to which information in the vCard pertains.

Value type: text

Special notes: The NAME property is used to convey the display name of the entity to which the directory information pertains. Its value is the displayable, presentation text associated with the source for the vCard, as specified in the SOURCE property.

Example:

```
NAME:Babs Jensen's Contact Information
```

5.1.5. KIND

Purpose: To specify the kind of object the vCard represents.

Value type: A single text value.

Special notes: The value may be one of: "individual" for a single person, "group" for a group of people, "org" for an organization, an x-name or an iana-token. If this property is absent, "individual" MUST be assumed as default.

Example:

This represents someone named Jane Doe working in the marketing department of the North American division of ABC Inc.

```
BEGIN:VCARD
VERSION:4.0
KIND:individual
FN:Jane Doe
ORG:ABC\, Inc.;North American Division;Marketing
END:VCARD
```

This represents the department itself, commonly known as ABC Marketing.

```
BEGIN:VCARD
VERSION:4.0
KIND:org
FN:ABC Marketing
ORG:ABC\, Inc.;North American Division;Marketing
END:VCARD
```

5.2. Identification Properties

These types are used to capture information associated with the identification and naming of the person or resource associated with the vCard.

5.2.1. FN

Purpose: To specify the formatted text corresponding to the name of the object the vCard represents.

Value type: A single text value.

Special notes: This property is based on the semantics of the X.520 Common Name attribute. The property MUST be present in the vCard object.

Example:

```
FN:Mr. John Q. Public\, Esq.
```

5.2.2. N

Purpose: To specify the components of the name of the object the vCard represents.

Value type: A single structured text value. Each component can have multiple values.

Special note: The structured type value corresponds, in sequence, to the Family Name, Given Name, Additional Names, Honorific Prefixes, and Honorific Suffixes. The text components are separated by the SEMI-COLON character (ASCII decimal 59). Individual text components can include multiple text values (e.g., multiple Additional Names) separated by the COMMA character (ASCII decimal 44). This type is based on the semantics of the X.520 individual name attributes. The property SHOULD be present in the vCard object when the name of the object the vCard represents follows the X.520 model.

Examples:

```
N:Public;John;Quinlan;Mr.;Esq.

N:Stevenson;John;Philip,Paul;Dr.;Jr.,M.D.,A.C.P.
```

5.2.3. NICKNAME

Purpose: To specify the text corresponding to the nickname of the object the vCard represents.

Value type: One or more text values separated by a COMMA character (ASCII decimal 44).

Special note: The nickname is the descriptive name given instead of or in addition to the one belonging to a person, place, or thing. It can also be used to specify a familiar form of a proper name specified by the FN or N types.

Examples:

```
NICKNAME:Robbie

NICKNAME:Jim,Jimmie
```

5.2.4. PHOTO

Purpose: To specify an image or photograph information that annotates some aspect of the object the vCard represents.

Encoding: The encoding MUST be reset to "b" using the ENCODING parameter in order to specify inline, encoded binary data. If the value is referenced by a URI value, then the default encoding is used and no explicit ENCODING parameter is needed.

Value type: A single value. The default is binary value. It can also be reset to uri value. The uri value can be used to specify a value outside of this MIME entity.

Special notes: This property SHOULD include the parameter "TYPE" to specify the graphic image format type. The TYPE parameter value MUST be an image media type as specified in [RFC4288]. The full media type name, including the "image/" prefix, should be used. However, implementations SHOULD be able to handle bare subtypes.

Example:

```
PHOTO;VALUE=uri:http://www.abc.com/pub/photos
 /jqpublic.gif


PHOTO;ENCODING=b;TYPE=image/jpeg:MIICajCCAdOgAwIBAgICBEUwDQYJKo
 ZIhvcNAQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENv
 bW11bmljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbi
 <...remainder of "B" encoded binary data...>
```

5.2.5. BDAY

Purpose: To specify the birth date of the object the vCard represents.

Value type: The default is a single date value. It can also be reset to a single date-time or text value.

Examples:

```
BDAY:1996-04-15

BDAY:1953-10-15T23:10:00Z

BDAY;VALUE=text:circa 1800
```

5.2.6. DDAY

Purpose: To specify the date of death of the object the vCard represents.

Value type: The default is a single date value. It can also be reset to a single date-time or text value.

5.2.7. BIRTH

Purpose: To specify the place of birth of the object the vCard represents.

Value type: A single text value.

Example:

```
BIRTH:Babies'R'Us Hospital
```

5.2.8. DEATH

Purpose: To specify the place of death of the object the vCard represents.

Value type: A single text value.

Example:

```
DEATH:Aboard the Titanic\, near Newfoundland
```

5.2.9. GENDER

Purpose: To specify the gender of the object the vCard represents.

Value type: A single text value.

Special notes: The value "M" stands for male while "F" stands for female.

Example:

```
GENDER:F
```

5.3. Delivery Addressing Properties

These types are concerned with information related to the delivery addressing or label for the vCard object.

5.3.1. ADR

Purpose: To specify the components of the delivery address for the vCard object.

Value type: A single structured text value, separated by the SEMI-COLON character (ASCII decimal 59).

Special notes: The structured type value consists of a sequence of address components. The component values MUST be specified in their corresponding position. The structured type value corresponds, in sequence, to the post office box; the extended address (e.g. apartment or suite number); the street address; the locality (e.g., city); the region (e.g., state or province); the postal code; the country name. When a component value is missing, the associated component separator MUST still be specified.

The text components are separated by the SEMI-COLON character (ASCII decimal 59). Where it makes semantic sense, individual text components can include multiple text values (e.g., a "street" component with multiple lines) separated by the COMMA character (ASCII decimal 44).

The type can include the type parameter "TYPE" to specify the delivery address type. The TYPE parameter values can include "home" to indicate a delivery address for a residence; "work" to indicate delivery address for a place of work; and "pref" to indicate the preferred delivery address when more than one address is specified. These type parameter values can be specified as a parameter list (i.e., "TYPE=home;TYPE=pref") or as a value list (i.e., "TYPE=home,pref"). This type is based on semantics of the X.520 geographical and postal addressing attributes. The default is "TYPE=work".

Example: In this example the post office box and the extended address are absent.

```
ADR;TYPE=home:;;123 Main Street;Any Town;CA;91921-1234
```

5.3.2. LABEL

Purpose: To specify the formatted text corresponding to delivery address of the object the vCard represents.

Value type: A single text value.

Special notes: The property value is formatted text that can be used to present a delivery address label for the vCard object. The type can include the type parameter "TYPE" to specify delivery label type. The TYPE parameter values can include "home" to indicate a delivery label for a residence; "work" to indicate delivery label for a place of work; and "pref" to indicate the preferred delivery label when more than one label is specified.

These type parameter values can be specified as a parameter list (i.e., "TYPE=home;TYPE=pref") or as a value list (i.e., "TYPE=home,pref"). The default is "TYPE=work".

Example: A multi-line address label.

```
LABEL;TYPE=home:Mr.John Q. Public\, Esq.\nMail Drop: TNE QB\n
 123 Main Street\nAny Town\, CA  91921-1234\nU.S.A.
```

5.4. Communications Properties

These properties are concerned with information associated with the way communications with the object the vCard represents are carried out.

5.4.1. TEL

Purpose: To specify the telephone number for telephony communication with the object the vCard represents.

Value type: A single phone-number value.

Special notes: The value of this property is specified in a canonical form in order to specify an unambiguous representation of the globally unique telephone endpoint. This property is based on the X.500 Telephone Number attribute.

The property can include the parameter "TYPE" to specify intended use for the telephone number. The TYPE parameter values can include: "home" to indicate a telephone number associated with a residence, "msg" to indicate the telephone number has voice messaging support, "work" to indicate a telephone number associated with a place of work, "pref" to indicate a preferred-use telephone number, "voice" to indicate a voice telephone number, "fax" to indicate a facsimile telephone number, "cell" to indicate a cellular telephone number, "video" to indicate a video conferencing telephone number, "pager" to indicate a paging device telephone number, "bbs" to indicate a bulletin board system telephone number, "modem" to indicate a MODEM connected telephone number, "car" to indicate a car-phone telephone number, "isdn" to indicate an ISDN service telephone number, "pcs" to indicate a personal communication services telephone number. The default type is "voice". These type parameter values can be specified as a parameter list (i.e., "TYPE=work;TYPE=voice") or as a value list (i.e., "TYPE=work,voice"). The default can be overridden to another set of values by specifying one or more alternate values. For example, the default TYPE of "voice" can be reset to a WORK and HOME, VOICE and FAX telephone number by the value list

"TYPE=work,home,voice,fax".

Example:

```
TEL;TYPE=work,voice,pref,msg:+1-213-555-1234
```

5.4.2. EMAIL

Purpose: To specify the electronic mail address for communication with the object the vCard represents.

Value type: A single text value.

Special notes: The type can include the type parameter "TYPE" to specify the format or preference of the electronic mail address. The TYPE parameter values can include: "internet" to indicate an Internet addressing type, "x400" to indicate a X.400 addressing type, "uri" to indicate a URI useable for electronic communication, "home" to indicate an address associated with a residence, "work" to indicate an address associated with a place of work, or "pref" to indicate a preferred-use email address when more than one is specified. Another IANA registered address type can also be specified. The default email type is "internet". A non-standard value can also be specified.

Type example:

```
EMAIL;TYPE=internet:jqpublic@xyz.dom1.com

EMAIL;TYPE=internet,pref:jane_doe@abc.com

EMAIL;TYPE=uri,work:http://example.com/contact.php
```

5.4.3. IMPP

Purpose: To specify the URI for instant messaging and presence protocol communications with the object the vCard represents.

Value type: A single URI. The type of the URI indicates the protocol that can be used for this contact.

Special notes: The property may include the type parameter "TYPE" to specify an intended use for the URI. The TYPE parameter values include one or more of the following:

* An indication of the type of communication for which this URI is appropriate. This can be a value of "personal" or "business".

* An indication of the location of a device associated with this URI. Values can be "home", "work", or "mobile".

* The value "pref" indicates this is a preferred address and has the same semantics as the "pref" value in a TEL property.

Example:

```
IMPP;TYPE=personal,pref:xmpp:alice@example.com
```

5.4.4. LANG

Purpose: To specify the language(s) that may be used for contacting the individual associated with the vCard.

Value type: A list of text values.

Special notes: The list is to be interpreted as defined in [RFC2616], Section 14.4, i.e. as the value of an Accept-Language HTTP header. This lets one specify preference among languages. Note that any SEMI-COLON character (ASCII decimal 59) must be escaped.

Example:

```
LANG:fr,en\;q=0.9
```

5.5. Geographical Properties

These properties are concerned with information associated with geographical positions or regions associated with the object the vCard represents.

5.5.1. TZ

Purpose: To specify information related to the time zone of the object the vCard represents.

Value type: The default is a single utc-offset value. It can also be reset to a single text value.

Special notes: The type value consists of a single value.

Type examples:

```
TZ:-05:00

TZ;VALUE=text:-05:00; EST; Raleigh/North America
```

```
;This example has a single value, not a structure text value.
```

5.5.2. GEO

Purpose: To specify information related to the global positioning of the object the vCard represents.

Value type: A single structured value consisting of two float values separated by the SEMI-COLON character (ASCII decimal 59).

Special notes: This property specifies information related to the global position of the object associated with the vCard. The value specifies latitude and longitude, in that order (i.e., "LAT LON" ordering). The longitude represents the location east and west of the prime meridian as a positive or negative real number, respectively. The latitude represents the location north and south of the equator as a positive or negative real number, respectively. The longitude and latitude values MUST be specified as decimal degrees and should be specified to six decimal places. This will allow for granularity within a meter of the geographical position. The text components are separated by the SEMI-COLON character (ASCII decimal 59). The simple formula for converting degrees-minutes-seconds into decimal degrees is:

```
decimal = degrees + minutes/60 + seconds/3600.
```

Example:

```
GEO:37.386013;-122.082932
```

5.6. Organizational Properties

These properties are concerned with information associated with characteristics of the organization or organizational units of the object the vCard represents.

5.6.1. TITLE

Purpose: To specify the job title, functional position or function of the object the vCard represents.

Value type: A single text value.

Special notes: This property is based on the X.520 Title attribute.

Example:

```
TITLE:Director\, Research and Development
```

5.6.2. ROLE

Purpose: To specify information concerning the role, occupation, or business category of the object the vCard represents.

Value type: A single text value.

Special notes: This property is based on the X.520 Business Category explanatory attribute. This property is included as an organizational type to avoid confusion with the semantics of the TITLE property and incorrect usage of that property when the semantics of this property is intended.

Example:

```
ROLE:Programmer
```

5.6.3. LOGO

Purpose: To specify a graphic image of a logo associated with the object the vCard represents.

Encoding: The encoding MUST be reset to "b" using the ENCODING parameter in order to specify inline, encoded binary data. If the value is referenced by a URI value, then the default encoding of 8bit is used and no explicit ENCODING parameter is needed.

Value type: A single value. The default is binary value. It can also be reset to uri value. The uri value can be used to specify a value outside of this MIME entity.

Special notes: This property SHOULD include the parameter "TYPE" to specify the graphic image format type. The TYPE parameter value MUST be an image media type as specified in [RFC4288]. The full media type name, including the "image/" prefix, should be used. However, implementations SHOULD be able to handle bare subtypes.

Example:

```
LOGO;VALUE=uri:http://www.abc.com/pub/logos/abccorp.jpg

LOGO;ENCODING=b;TYPE=image/jpeg:MIICajCCAdOgAwIBAgICBEUwDQYJKoZ
 AQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bm
 ljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbiBTeXN0
 <...the remainder of "B" encoded binary data...>
```

5.6.4. AGENT

Purpose: To specify information about another person who will act on behalf of the individual or resource associated with the vCard.

Value type: The default is a single vcard value. It can also be reset to either a single text or uri value. The text value can be used to specify textual information. The uri value can be used to specify information outside of this MIME entity.

Special notes: This property typically is used to specify an area administrator, assistant, or secretary for the individual associated with the vCard. A key characteristic of the AGENT property is that it represents somebody or something that is separately addressable.

Example:

```
AGENT;VALUE=uri:
 CID:JQPUBLIC.part3.960129T083020.xyzMail@host3.com


AGENT:BEGIN:VCARD\nFN:Susan Thomas\nTEL:+1-919-555-
 1234\nEMAIL\;INTERNET:sthomas@host.com\nEND:VCARD\n
```

5.6.5. ORG

Purpose: To specify the organizational name and units associated with the vCard.

Value type: A single structured text value consisting of components separated the SEMI-COLON character (ASCII decimal 59).

Special notes: The property is based on the X.520 Organization Name and Organization Unit attributes. The property value is a structured type consisting of the organization name, followed by one or more levels of organizational unit names.

Example: A property value consisting of an organizational name, organizational unit #1 name and organizational unit #2 name.

```
ORG:ABC\, Inc.;North American Division;Marketing
```

5.7. Explanatory Properties

These properties are concerned with additional explanations, such as that related to informational notes or revisions specific to the vCard.

5.7.1. CATEGORIES

Purpose: To specify application category information about the vCard.

Value type: One or more text values separated by a COMMA character (ASCII decimal 44).

Example:

```
CATEGORIES:TRAVEL AGENT

CATEGORIES:INTERNET,IETF,INDUSTRY,INFORMATION TECHNOLOGY
```

5.7.2. NOTE

Purpose: To specify supplemental information or a comment that is associated with the vCard.

Value type: A single text value.

Special notes: The property is based on the X.520 Description attribute.

Example:

```
NOTE:This fax number is operational 0800 to 1715
  EST\, Mon-Fri.
```

5.7.3. PRODID

Purpose: To specify the identifier for the product that created the vCard object.

Type value: A single text value.

Special notes: Implementations SHOULD use a method such as that specified for Formal Public Identifiers in [ISO9070] or for Universal Resource Names in [RFC3406] to assure that the text value is unique.

Example:

```
PRODID:-//ONLINE DIRECTORY//NONSGML Version 1//EN
```

5.7.4. REV

Purpose: To specify revision information about the current vCard.

Value type: The default is a single date-time value. Can also be reset to a single date value.

Special notes: The value distinguishes the current revision of the information in this vCard for other renditions of the information.

Example:

```
REV:1995-10-31T22:27:10Z

REV:1997-11-15
```

5.7.5. SORT-STRING

Purpose: To specify the family name or given name text to be used for national-language-specific sorting of the FN and N types.

Value type: A single text value.

Special notes: The sort string is used to provide family name or given name text that is to be used in locale- or national-language- specific sorting of the formatted name and structured name types. Without this information, sorting algorithms could incorrectly sort this vCard within a sequence of sorted vCards. When this property is present in a vCard, then this family name or given name value is used for sorting the vCard.

Examples: For the case of family name sorting, the following examples define common sort string usage with the FN and N properties.

```
FN:Rene van der Harten
N:van der Harten;Rene;J.;Sir;R.D.O.N.
SORT-STRING:Harten

FN:Robert Pau Shou Chang
N:Pau;Shou Chang;Robert
SORT-STRING:Pau

FN:Osamu Koura
N:Koura;Osamu
SORT-STRING:Koura

FN:Oscar del Pozo
N:del Pozo Triscon;Oscar
SORT-STRING:Pozo

FN:Chistine d'Aboville
N:d'Aboville;Christine
SORT-STRING:Aboville
```

5.7.6. SOUND

Purpose: To specify a digital sound content information that annotates some aspect of the vCard. By default this property is used to specify the proper pronunciation of the name property value of the vCard.

Encoding: The encoding MUST be reset to "b" using the ENCODING parameter in order to specify inline, encoded binary data. If the value is referenced by a URI value, then the default encoding of 8bit is used and no explicit ENCODING parameter is needed.

Value type: A single value. The default is binary value. It can also be reset to uri value. The uri value can be used to specify a value outside of this MIME entity.

Special notes: This property SHOULD include the parameter "TYPE" to specify the audio format type. The TYPE parameter value MUST be an audio media type as specified in [RFC4288]. The full media type name, including the "audio/" prefix, should be used. However, implementations SHOULD be able to handle bare subtypes.

Example:

```
SOUND;TYPE=audio/basic;VALUE=uri:CID:JOHNQPUBLIC.part8.
 19960229T080000.xyzMail@host1.com

SOUND;TYPE=audio/basic;ENCODING=b:MIICajCCAdOgAwIBAgICBEUwDQYJK
 AQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bm
 ljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbiBTeXN0
 <...the remainder of "B" encoded binary data...>
```

5.7.7. UID

Purpose: To specify a value that represents a globally unique identifier corresponding to the individual or resource associated with the vCard.

Value type: A single text value.

Special notes: The type is used to uniquely identify the object that the vCard represents.

The type can include the type parameter "TYPE" to specify the format of the identifier. The TYPE parameter value should be an IANA registered identifier format. The value can also be a non-standard format.

Example:

```
UID:19950401-080045-40000F192713-0052
```

5.7.8. URL

Purpose: To specify a uniform resource locator associated with the object that the vCard refers to.

Value type: A single uri value.

Example:

```
URL:http://www.swbyps.restaurant.french/~chezchic.html
```

5.7.9. VERSION

Purpose: To specify the version of the vCard specification used to format this vCard.

Value type: A single text value.

Special notes: The property MUST be present in the vCard object. The value MUST be "4.0" if the vCard corresponds to this specification.

Example:

```
VERSION:4.0
```

5.8. Security Properties

These properties are concerned with the security of communication pathways or access to the vCard.

5.8.1. CLASS

Purpose: To specify the access classification for a vCard object.

Value type: A single text value.

Special notes: An access classification is only one component of the general security model for a directory service. The classification attribute provides a method of capturing the intent of the owner for general access to information described by the vCard object.

Examples:

```
CLASS:PUBLIC

CLASS:PRIVATE

CLASS:CONFIDENTIAL
```

5.8.2. KEY

Purpose: To specify a public key or authentication certificate associated with the object that the vCard represents.

Encoding: The encoding MUST be reset to "b" using the ENCODING parameter in order to specify inline, encoded binary data. If the value is a text value, then the default encoding of 8bit is used and no explicit ENCODING parameter is needed.

Value type: A single value. The default is binary. It can also be reset to text value. The text value can be used to specify a text key.

Special notes: The property can also include the parameter TYPE to specify the public key or authentication certificate format. This parameter should specify an IANA registered public key or authentication certificate format. It can also specify a non-standard format.

Special notes: This property SHOULD include the parameter "TYPE" to specify the public key or authentication certificate format. The TYPE parameter value MUST be a media type as specified in [RFC4288].

Example:

```
KEY;TYPE=application/pgp-keys;ENCODING=b:mQGiBEbEPUsRBACBF0RSIN
 mGutdM+KSAl7HMzwXHaLbvEOyu8At80I8qGejhzWowKbfem3X0m68Y/vhb+J2g
 7q11KHpnEdNb67uZaj9nTQ09Q+UFtH25qD/Afn3+9bOJQaPjAUYzXu3vD/xmN8
 <...remainder of "B" encoded binary data...>
```

5.9. Extended Properties and Parameters

The properties and parameters defined by this document can be extended. Non-standard, private properties and parameters with a name starting with "X-" may be defined bilaterally between two cooperating agents without outside registration or standardization.

6. Formal Grammar

The following formal grammar is provided to assist developers in building parsers for the vCard.

This syntax is written according to the form described in [RFC5234], but it references just this small subset of [RFC5234] literals:

```
;****************************************
; Commonly Used Literal Definition
;****************************************

ALPHA          = %x41-5A / %x61-7A
     ; Latin Capital Letter A-Latin Capital Letter Z /
     ; Latin Small Letter a-Latin Small Letter z
CHAR           = %x01-7F
     ; Any C0 Controls and Basic Latin, excluding NULL from
     ; Code Charts, pages 7-6 through 7-9 in [UNICODE]

CR             = %x0D
     ; Carriage Return

LF             = %0A
```

```
         ; Line Feed

    CRLF         = CR LF
         ; Internet standard newline

    ;CTL         = %x00-1F / %x7F
         ; Controls. Not used, but referenced in comments.

    DIGIT        = %x30-39
         ; Digit Zero-Digit Nine

    DQUOTE       = %x22
         ; Quotation Mark

    HTAB         = %x09
         ; Horizontal Tabulation

    SP           = %x20
         ; space

    VCHAR        = %x21-7E
         ; Visible (printing) characters

    WSP          = SP / HTAB
         ; White Space

    ;****************************************
    ; Basic vCard Definition
    ;****************************************

    vcard_entity = 1*(vcard)

    vcard        = "BEGIN" ":" "VCARD" 1*CRLF
                   1*(contentline)
         ;A vCard object MUST include the VERSION, FN and N types.
                   "END" ":" "VCARD" 1*CRLF

    contentline  = name *(";" param ) ":" value CRLF
         ; When parsing a content line, folded lines must first
         ; be unfolded according to the unfolding procedure
         ; described above. When generating a content line, lines
         ; longer than 75 characters SHOULD be folded according to
         ; the folding procedure described in [MIME DIR].

    name         = iana-token / x-name
         ; Parsing of the param and value is
         ; based on the "name" or type identifier
         ; as defined in ABNF sections below
```

```
    iana-token   = 1*(ALPHA / DIGIT / "-")
         ; vCard type or parameter identifier registered with IANA

    x-name       = "X-" 1*(ALPHA / DIGIT / "-")
         ; Reserved for non-standard use

    param        = param-name "=" param-value *("," param-value)

    param-name   = iana-token / x-name

    param-value  = ptext / quoted-string

    ptext        = *SAFE-CHAR

    value        = *VALUE-CHAR

    quoted-string = DQUOTE QSAFE-CHAR DQUOTE

    NON-ASCII    = %x80-FF
         ; Use is restricted by outer MIME object (UTF-8 preferred)

    QSAFE-CHAR   = WSP / %x21 / %x23-7E / NON-ASCII
         ; Any character except CTLs, DQUOTE

    SAFE-CHAR    = WSP / %x21 / %x23-2B / %x2D-39 / %x3C-7E / NON-ASCII
         ; Any character except CTLs, DQUOTE, ";", ":", ","

    VALUE-CHAR   = WSP / VCHAR / NON-ASCII
         ; Any textual character

    ;****************************************
    ; vCard Type Definition
    ;
    ; Provides type-specific definitions for how the
    ; "value" and "param" are defined.
    ;****************************************
    ;For name="NAME"
    param        = ""
         ; No parameters allowed

    value        = text-value

    ;For name="KIND"
    param        = ""
         ; No parameters allowed

    value        = kind-value
```

```
    kind-value   = "individual" / "group" / "org" / x-name / iana-token

    ;For name="PROFILE"
    param        = ""
         ; No parameters allowed

    value        = text-value
         ; Value MUST be the case insensitive value "VCARD

    ;For name="SOURCE"
    param        = source-param
         ; Only source parameters allowed

    value        = uri

    source-param = ("VALUE" "=" "uri")
                 / (x-name "=" *SAFE-CHAR)

    ;For name="FN"
    ;This type MUST be included in a vCard object.
    param        = text-param
         ; Text parameters allowed

    value        = text-value

    ;For name="N"
    ;This type MUST be included in a vCard object.

    param        = text-param
         ; Text parameters allowed

    value        = n-value

    n-value      = 0*4(text-value *("," text-value) ";")
                   text-value *("," text-value)
         ; Family; Given; Middle; Prefix; Suffix.
         ; Example: Public;John;Quincy,Adams;Reverend Dr. III

    ;For name="NICKNAME"
    param        = text-param
         ; Text parameters allowed
    value        = text-value-list

    ;For name="PHOTO"
    param        = img-inline-param
         ; Only image parameters allowed

    param        =/ img-refer-param
```

```
         ; Only image parameters allowed

    value        = img-inline-value
         ; Value and parameter MUST match

    value        =/ img-refer-value
         ; Value and parameter MUST match

    ;For name="BDAY"
    param        = ("VALUE" "=" "date")
         ; Only value parameter allowed

    param        =/ ("VALUE" "=" "date-time")
         ; Only value parameter allowed

    value        = date-value
         ; Value MUST match value type

    value        =/ date-time-value
         ; Value MUST match value type

    ;For name="ADR"
    param        = adr-param / text-param
         ; Only adr and text parameters allowed

    value        = adr-value

    ;For name="LABEL"
    param        = adr-param / text-param
         ; Only adr and text parameters allowed

    value        = text-value

    ;For name="TEL"
    param        = tel-param
         ; Only tel parameters allowed

    value        = phone-number-value

    tel-param    = "TYPE" "=" tel-type *("," tel-type)
    tel-type     = "HOME" / "WORK" / "PREF" / "VOICE" / "FAX" / "MSG"
                 / "CELL" / "PAGER" / "BBS" / "MODEM" / "CAR" / "ISDN"
                 / "VIDEO" / "PCS" / iana-token / x-name
         ; Values are case insensitive

    ;For name="EMAIL"
    param        = email-param
         ; Only email parameters allowed
```

```
    value        = text-value

    email-param  = "TYPE" "=" email-type ["," "PREF"]
         ; Value is case insensitive

    email-type   = "INTERNET" / "X400" / iana-token / "X-" word
         ; Values are case insensitive

    ;For name="TZ"
    param        = ""
         ; No parameters allowed

    value        = utc-offset-value

    ;For name="GEO"
    param        = ""
         ; No parameters allowed

    value        = float-value ";" float-value

    ;For name="TITLE"
    param        = text-param
         ; Only text parameters allowed

    value        = text-value

    ;For name="ROLE"
    param        = text-param
         ; Only text parameters allowed

    value        = text-value

    ;For name="LOGO"
    param        = img-inline-param / img-refer-param
         ; Only image parameters allowed

    value        = img-inline-value / img-refer-value
         ; Value and parameter MUST match

    ;For name="AGENT"
    param        = agent-inline-param

    param        =/ agent-refer-param

    param        =/ text-param

    value        = agent-inline-value
         ; Value and parameter MUST match
```

```
    value        =/ agent-refer-value
         ; Value and parameter MUST match

    value        =/ text-value
         ; Value and parameter MUST match

    agent-inline-param = ""
         ; No parameters allowed

    agent-refer-param = "VALUE" "=" "uri"
         ; Only value parameter allowed

    agent-inline-value = text-value
         ; Value MUST be a valid vCard object

    agent-refer-value = uri
         ; URI MUST refer to valid vCard object

    ;For name="ORG"

    param        = text-param
         ; Only text parameters allowed

    value        = org-value

    org-value    = *(text-value ";") text-value
         ; First is Organization Name, remainder are Organization Units.

    ;For name="CATEGORIES"
    param        = text-param
         ; Only text parameters allowed

    value        = text-value-list

    ;For name="NOTE"
    param        = text-param
         ; Only text parameters allowed
    value        = text-value

    ;For name="PRODID"
    param        = ""
         ; No parameters allowed

    value        = text-value

    ;For name="REV"
    param        = ["VALUE" =" "date-time"]
         ; Only value parameters allowed. Values are case insensitive.
```

```
    param        =/ "VALUE" =" "date"
         ; Only value parameters allowed. Values are case insensitive.

    value        = date-time-value

    value        =/ date-value

    ;For name="SORT-STRING"
    param        = text-param
         ; Only text parameters allowed

    value        = text-value

    ;For name="SOUND"
    param        = snd-inline-param
         ; Only sound parameters allowed

    param        =/ snd-refer-param
         ; Only sound parameters allowed

    value        = snd-line-value
         ; Value MUST match value type

    value        =/ snd-refer-value
         ; Value MUST match value type

    snd-inline-value     = binary-value CRLF
         ; Value MUST be "b" encoded audio content

    snd-inline-param     = ("VALUE" "=" "binary"])
                         / ("ENCODING" "=" "b")
                         / ("TYPE" "=" *SAFE-CHAR)
         ; Value MUST be an IANA registered audio type

    snd-refer-value      = uri
         ; URI MUST refer to audio content of given type
    snd-refer-param      = ("VALUE" "=" "uri")
                         / ("TYPE" "=" word)
         ; Value MUST be an IANA registered audio type

    ;For name="UID"
    param        = "TYPE" "=" (iana-token / x-name)
         ;TYPE value should be an IANA registered identifier format

    value        = text-value

    ;For name="URL"
    param        = ""
```

```
     ; No parameters allowed

value        = uri

;For name="VERSION"
;This type MUST be included in a vCard object.
param        = ""
     ; No parameters allowed

value        = text-value
     ; Value MUST be "3.0"

;For name="CLASS"
param        = ""
     ; No parameters allowed

value        = "PUBLIC" / "PRIVATE" / "CONFIDENTIAL"
             / iana-token / x-name
     ; Value are case insensitive

;For name="KEY"
param        = key-txt-param
     ; Only value and type parameters allowed

param        =/ key-bin-param
     ; Only value and type parameters allowed

value        = text-value

value        =/ binary-value

key-txt-param = "TYPE" "=" keytype

key-bin-param = ("TYPE" "=" keytype)
              / ("ENCODING" "=" "b")
     ; Value MUST be a "b" encoded key or certificate
keytype      = param-value
     ; Type MUST be a media type as defined in RFC 4288

;For name="X-" non-standard type
param        = text-param / (x-name "=" param-value)
     ; Only text or non-standard parameters allowed

value        = text-value

;****************************************
; vCard Commonly Used Parameter Definition
;****************************************
```

```
    text-param   = ("VALUE" "=" "ptext")
                 / ("LANGUAGE" "=" langval)
                 / (x-name "=" param-value)

    langval      = <a language string as defined in [RFC4646]>

    img-inline-value     = binary-value
         ;Value MUST be "b" encoded image content

    img-inline-param

    img-inline-param     = ("VALUE" "=" "binary")
                         / ("ENCODING" "=" "b")
                         / ("TYPE" "=" param-value
         ;TYPE value MUST be an image media type as defined in RFC 4288

    img-refer-value = uri
         ;URI MUST refer to image content of given type

    img-refer-param      = ("VALUE" "=" "uri")
                         / ("TYPE" "=" param-value)
         ;TYPE value MUST be an image media type as defined in RFC 4288

    adr-param    = ("TYPE" "=" adr-type *("," adr-type))
                 / (text-param)

    adr-type     = "home" / "work" / "pref" / iana-token / x-name

    adr-value    = 0*6(text-value ";") text-value
         ; PO Box, Extended Address, Street, Locality, Region, Postal
         ; Code, Country Name
    ;****************************************
    ; vCard Type Value Definition
    ;****************************************

    text-value-list      = 1*text-value *("," 1*text-value)

    text-value   = *(SAFE-CHAR / ":" / DQUOTE / ESCAPED-CHAR)

    ESCAPED-CHAR = "\\" / "\;" / "\," / "\n" / "\N")
         ; \\ encodes \, \n or \N encodes newline
         ; \; encodes ;, \, encodes ,

    binary-value = <A "b" encoded text value as defined in [RFC2047]>

    date-value   = <A single date value as defined in [RFC2425]>

    time-value   = <A single time value as defined in [RFC2425]>
```

```
date-time-value = <A single date-time value as defined in [RFC2425]>

float-value  = <A single float value as defined in [RFC2425]>

phone-number-value = <A single text value as defined in [CCITT
                      E.163] and [CCITT.X121.1988]>

uri-value    = <A uri value as defined in [RFC2425]>

utc-offset-value = ("+" / "-") time-hour ":" time-minute
time-hour    = 2DIGIT               ;00-23
time-minute  = 2DIGIT               ;00-59
```

7. Example: Authors' vCards

```
BEGIN:VCARD
VERSION:4.0
FN:Pete Resnick
N:Resnick;Pete;;;
GENDER:M
ORG:QUALCOMM Incorporated
ADR;TYPE=work:;;5775 Morehouse Drive;San Diego;CA;92121-1714;US
TEL;TYPE=voice:+1-858-651-4478
EMAIL;TYPE=internet:presnick@qualcomm.com
URL:http://www.qualcomm.com/~presnick/
END:VCARD


BEGIN:VCARD
VERSION:4.0
FN:Simon Perreault
N:Perreault;Simon;;;ing. jr.,M.Sc.
BDAY:1983-02-03
GENDER:M
ORG:Viagenie
ADR;TYPE=work:;;2600 boul. Laurier\, suite 625;
 Quebec;QC;G1V 4W1;Canada
TEL;TYPE=voice,work:+1-418-656-9254
TEL;TYPE=fax,work:+1-418-656-9257
EMAIL;TYPE=internet,work:simon.perreault@viagenie.ca
GEO:46.772673,-71.282945
CLASS:PUBLIC
KEY;VALUE=uri:http://www.viagenie.ca/simon.perreault/simon.asc
END:VCARD
```

8. Security Considerations

- Internet mail is subject to many well known security attacks, including monitoring, replay, and forgery. Care should be taken by any directory service in allowing information to leave the scope of the service itself, where any access controls can no longer be guaranteed. Applications should also take care to display directory data in a "safe" environment (e.g., PostScript-valued types).

- vCards can carry cryptographic keys or certificates, as described in Section 5.8.2.

- Section 5.8.1 specifies a desired security classification policy for a particular vCard. That policy is not enforced in any way.

- The vCard objects have no inherent authentication or privacy, but can easily be carried by any security mechanism that transfers MIME objects with authentication or privacy. In cases where threats of "spoofed" vCard information is a concern, the vCard SHOULD BE transported using one of these secure mechanisms.

- The information in a vCard may become out of date. In cases where the vitality of data is important to an originator of a vCard, the "URL" type described in Section 5.7.8 SHOULD BE specified. In addition, the "REV" type described in section Section 5.7.4 can be specified to indicate the last time that the vCard data was updated.

9. Acknowledgements

The authors would like to thank Frank Dawson and Tim Howes, the original authors of [RFC2425] and [RFC2426], as well as the following individuals who have participated in the drafting, review and discussion of this memo:

Marc Blanchet, Darryl Champagne, Cyrus Daboo, Javier Godoy, Mark Paterson, and Julien Reschke.

10. References

10.1. Normative References

[CCITT.E163.1988] International Telephone and Telegraph Consultative Committee, "Numbering Plan for the International Telephone Service", CCITT Recommendation E.163, 1988.

[CCITT.X121.1988] International Telephone and Telegraph Consultative Committee, "International Numbering Plan for the Public Data Networks", CCITT Recommendation X.121, 1988.

[CCITT.X520.1988] International International Telephone and Telegraph Consultative Committee, "Information Technology - Open Systems Interconnection - The Directory: Selected Attribute Types", CCITT Recommendation X.520, November 1988.

[CCITT.X521.1988] International International Telephone and Telegraph Consultative Committee, "Information Technology - Open Systems Interconnection - The Directory: Selected Object Classes", CCITT Recommendation X.521, November 1988.

[ISO.8601.1988] International Organization for Standardization, "Data elements and interchange formats - Information interchange - Representation of dates and times", ISO Standard 8601, June 1988.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

[RFC2047] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2425] Howes, T., Smith, M., and F. Dawson, "A MIME Content-Type for Directory Information", RFC 2425, September 1998.

[RFC2426] Dawson, F. and T. Howes, "vCard MIME Directory Profile", RFC 2426, September 1998.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol --

HTTP/1.1", RFC 2616, June 1999.

[RFC2822] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[RFC2978] Freed, N. and J. Postel, "IANA Charset Registration Procedures", BCP 19, RFC 2978, October 2000.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and Registration Procedures", BCP 13, RFC 4288, December 2005.

[RFC4646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 4646, September 2006.

[RFC4770] Jennings, C. and J. Reschke, Ed., "vCard Extensions for Instant Messaging (IM)", RFC 4770, January 2007.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[oldreference_UNICODE] The International Organization for Standardization, "The Unicode Standard - Version 2.0", The Unicode Consortium", July 1996.

[oldreference_VCARD] Internet Mail Consortium, "vCard - The Electronic Business Card Version 2.1", September September.

10.2. Informative References

[ISO9070] The International Organization for Standardization, "ISO 9070, Information Processing - SGML support facilities -

Registration Procedures for Public Text Owner Identifiers", April 1991.

[RFC3406] Daigle, L., van Gulik, D., Iannella, R., and P. Faltstrom, "Uniform Resource Names (URN) Namespace Definition Mechanisms", BCP 66, RFC 3406, October 2002.

Appendix A. Differences from RFCs 2425 and 2426

This appendix contains a list of changes that have been made in the vCard specification from RFCs 2425 and 2426.

A.1. New Structure

- [RFC2425] and [RFC2426] have been merged. Initially [RFC2425] was intended to be extensible but only 2426 ever extended it.
- vCard is now not only a MIME type but a stand-alone format.
- A proper MIME type registration form has been included.
- UTF-8 is now the default character set.

A.2. Removed Features

- The group construct (i.e. GROUP.PROPERTY:...) no longer exists.
- The CONTEXT and CHARSET parameters are no more.
- The MAILER property is no more.
- The "intl", "dom", "postal", and "parcel" TYPE parameter values for the ADR and LABEL properties have been removed.

A.3. New Properties and Parameters

- The KIND, GENDER, LANG, DDAY, BIRTH, and DEATH properties have been added.
- [RFC4770], which defines the IMPP property, has been merged in.
- The "work", "home", and "uri" TYPE parameter values for the EMAIL property have been added.

A.4. Other Changes

- The N property is no longer mandatory.

Appendix B. Change Log (to be removed by RFC Editor prior to publication)

B.1. Changes in -01

- Removed reference to RFC 2234.
- Fixed errata from http://www.rfc-editor.org/errata_search.php/doc/html/rfc2426.
- Removed passage referring to RFC 2425 profiles.
- Renamed Section 5.4 from "Telecommunications Adressing Properties" to "Communications Properties.
- Added Appendix A and Appendix B.
- Added reference to [RFC4770].
- Removed the group construct.
- Made the N property no longer mandatory.
- Added the KIND property.
- Clarified meaning of TYPE parameter value for PHOTO, LOGO, KEY, and SOUND.
- Removed the CONTEXT parameter.
- Removed the MAILER property.
- Made reference to [ISO9070] informative.
- Removed "intl", "dom", "postal", and "parcel" TYPE parameter values for the ADR and LABEL properties.
- Clarified meaning of "extended address" ADR field.
- Mentioned [RFC3406] as another method of generating PRODID values.
- Updated obsolete references.

- Allowed BDAY and DDAY value types to be text values for fuzzy dates.

- Removed the CHARSET property. Now the encoding is always UTF-8, except when overridden by the Content-Type (which is considered a compatibility feature).

Authors' Addresses

Peter W. Resnick
QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121-1714
US

Phone: +1 858 651 4478
EMail: presnick@qualcomm.com
URI: http://www.qualcomm.com/~presnick/

Simon Perreault
Viagenie
2600 boul. Laurier, suite 625
Quebec, QC G1V 4W1
Canada

Phone: +1 418 656 9254
EMail: simon.perreault@viagenie.ca
URI: http://www.viagenie.ca

Full Copyright Statement

Intellectual Property

Acknowledgements

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA). This document was produced using xml2rfc v1.32 (of http://xml.resource.org/) from a source in RFC-2629 XML format.