```
Internet Engineering Task Force (IETF)                       S. Perreault
Request for Comments: 6350                                        Viagenie
Obsoletes: 2425, 2426, 4770                                   August 2011
Updates: 2739
Category: Standards Track
ISSN: 2070-1721
```

vCard Format Specification

Abstract

   This document defines the vCard data format for representing and
   exchanging a variety of information about individuals and other
   entities (e.g., formatted and structured name and delivery addresses,
   email address, multiple telephone numbers, photograph, logo, audio
   clips, etc.). This document obsoletes RFCs 2425, 2426, and 4770, and
   updates RFC 2739.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6350.

Copyright Notice

Table of Contents

1. Introduction

Electronic address books have become ubiquitous. Their increased
presence on portable, connected devices as well as the diversity of
platforms that exchange 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.

A high-level overview of the differences from RFCs 2425 and 2426 can
be found in Appendix A.

2. Conventions

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

3. vCard Format Specification

The text/vcard MIME content type (hereafter known as "vCard"; see
Section 10.1) 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.

3.1. Charset

The charset (see [RFC3536] for internationalization terminology) for
vCard is UTF-8 as defined in [RFC3629]. There is no way to override
this. It is invalid to specify a value other than "UTF-8" in the
"charset" MIME parameter (see Section 10.1).

3.2. Line Delimiting and Folding

Individual lines within vCard are delimited by the [RFC5322] line
break, which is a CRLF sequence (U+000D followed by U+000A). Long
logical lines of text can be split into a multiple-physical-line
representation using the following folding technique. Content lines
SHOULD be folded to a maximum width of 75 octets, excluding the line
break. Multi-octet characters MUST remain contiguous. The rationale
for this folding process can be found in [RFC5322], Section 2.1.1.

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 (U+0020) or horizontal tab
(U+0009)). The folded line MUST contain at least one character. Any
sequence of CRLF followed immediately by a single white space

character is ignored (removed) when processing the content type. For
example, the line:

    NOTE:This is a long description that exists on a long line.

can be represented as:

    NOTE:This is a long description
      that exists on a long line.

It could also be represented as:

    NOTE: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 (U+0009) or SPACE
(U+0020)) as equivalent to no characters at all (i.e., the CRLF and
single white space character are removed).

    Note: It is possible for very simple implementations to generate
    improperly folded lines in the middle of a UTF-8 multi-octet
    sequence. For this reason, implementations SHOULD unfold lines in
    such a way as to properly restore the original sequence.

    Note: Unfolding is done differently than in [RFC5322]. Unfolding
    in [RFC5322] only removes the CRLF, not the space following it.

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.

3.3. ABNF Format Definition

The following ABNF uses the notation of [RFC5234], which also defines
CRLF, WSP, DQUOTE, VCHAR, ALPHA, and DIGIT.

vcard-entity = 1*vcard

vcard = "BEGIN:VCARD" CRLF
        "VERSION:4.0" CRLF
        1*contentline
        "END:VCARD" CRLF
  ; A vCard object MUST include the VERSION and FN properties.
  ; VERSION MUST come immediately after BEGIN:VCARD.

```
contentline = [group "."] name *(";" param) ":" value CRLF
  ; When parsing a content line, folded lines must first
  ; be unfolded according to the unfolding procedure
  ; described in Section 3.2.
  ; When generating a content line, lines longer than 75
  ; characters SHOULD be folded according to the folding
  ; procedure described in Section 3.2.

group = 1*(ALPHA / DIGIT / "-")
name  = "SOURCE" / "KIND" / "FN" / "N" / "NICKNAME"
      / "PHOTO" / "BDAY" / "ANNIVERSARY" / "GENDER" / "ADR" / "TEL"
      / "EMAIL" / "IMPP" / "LANG" / "TZ" / "GEO" / "TITLE" / "ROLE"
      / "LOGO" / "ORG" / "MEMBER" / "RELATED" / "CATEGORIES"
      / "NOTE" / "PRODID" / "REV" / "SOUND" / "UID" / "CLIENTPIDMAP"
      / "URL" / "KEY" / "FBURL" / "CALADRURI" / "CALURI" / "XML"
      / iana-token / x-name
  ; Parsing of the param and value is based on the "name" as
  ; defined in ABNF sections below.
  ; Group and name are case-insensitive.

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 = language-param / value-param / pref-param / pid-param
      / type-param / geo-parameter / tz-parameter / sort-as-param
      / calscale-param / any-param
  ; Allowed parameters depend on property name.

param-value = *SAFE-CHAR / DQUOTE *QSAFE-CHAR DQUOTE

any-param  = (iana-token / x-name) "=" param-value *("," param-value)

NON-ASCII = UTF8-2 / UTF8-3 / UTF8-4
  ; UTF8-{2,3,4} are defined in [RFC3629]

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

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

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

A line that begins with a white space character is a continuation of
the previous line, as described in Section 3.2. 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 [RFC5322], 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. Parameter values that are not explicitly defined as
being case-sensitive are case-insensitive. Based on experience with
vCard 3 interoperability, it is RECOMMENDED that property and
parameter names be upper-case on output.

The group construct is used to group related properties together.
The group name is a syntactic convention used to indicate that all
property names prefaced with the same group name SHOULD be grouped
together when displayed by an application. It has no other
significance. Implementations that do not understand or support
grouping MAY simply strip off any text before a "." to the left of
the type name and present the types and values as normal.

Property cardinalities are indicated using the following notation,
which is based on ABNF (see [RFC5234], Section 3.6):

```
+-------------+----------------------------------------------------+
| Cardinality | Meaning                                            |
+-------------+----------------------------------------------------+
|      1      | Exactly one instance per vCard MUST be present.    |
|     *1      | Exactly one instance per vCard MAY be present.     |
|     1*      | One or more instances per vCard MUST be present.   |
|      *      | One or more instances per vCard MAY be present.    |
+-------------+----------------------------------------------------+
```

Properties defined in a vCard instance may have multiple values
depending on the property cardinality. 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).

3.4. Property Value Escaping

Some properties may contain one or more values delimited by a COMMA
character (U+002C). Therefore, a COMMA character in a value MUST be
escaped with a BACKSLASH character (U+005C), even for properties that
don't allow multiple instances (for consistency).

Some properties (e.g., N and ADR) comprise multiple fields delimited
by a SEMICOLON character (U+003B). Therefore, a SEMICOLON in a field
of such a "compound" property MUST be escaped with a BACKSLASH
character. SEMICOLON characters in non-compound properties MAY be
escaped. On input, an escaped SEMICOLON character is never a field
separator. An unescaped SEMICOLON character may be a field
separator, depending on the property in which it appears.

Furthermore, some fields of compound properties may contain a list of
values delimited by a COMMA character. Therefore, a COMMA character
in one of a field's values MUST be escaped with a BACKSLASH
character, even for fields that don't allow multiple values (for
consistency). Compound properties allowing multiple instances MUST
NOT be encoded in a single content line.

Finally, BACKSLASH characters in values MUST be escaped with a
BACKSLASH character. NEWLINE (U+000A) characters in values MUST be
encoded by two characters: a BACKSLASH followed by either an 'n'
(U+006E) or an 'N' (U+004E).

In all other cases, escaping MUST NOT be used.

4. Property Value Data Types

Standard value types are defined below.

```
   value = text
         / text-list
         / date-list
         / time-list
         / date-time-list
         / date-and-or-time-list
         / timestamp-list
         / boolean
         / integer-list
         / float-list
         / URI                 ; from Section 3 of [RFC3986]
         / utc-offset
         / Language-Tag
         / iana-valuespec
      ; Actual value type depends on property name and VALUE parameter.
```

```
text = *TEXT-CHAR

TEXT-CHAR = "\\" / "\," / "\n" / WSP / NON-ASCII
          / %x21-2B / %x2D-5B / %x5D-7E
   ; Backslashes, commas, and newlines must be encoded.

component = "\\" / "\," / "\;" / "\n" / WSP / NON-ASCII
          / %x21-2B / %x2D-3A / %x3C-5B / %x5D-7E
list-component = component *("," component)

text-list            = text             *("," text)
date-list            = date             *("," date)
time-list            = time             *("," time)
date-time-list       = date-time        *("," date-time)
date-and-or-time-list = date-and-or-time *("," date-and-or-time)
timestamp-list       = timestamp        *("," timestamp)
integer-list         = integer          *("," integer)
float-list           = float            *("," float)

boolean = "TRUE" / "FALSE"
integer = [sign] 1*DIGIT
float   = [sign] 1*DIGIT ["." 1*DIGIT]

sign = "+" / "-"

year   = 4DIGIT  ; 0000-9999
month  = 2DIGIT  ; 01-12
day    = 2DIGIT  ; 01-28/29/30/31 depending on month and leap year
hour   = 2DIGIT  ; 00-23
minute = 2DIGIT  ; 00-59
second = 2DIGIT  ; 00-58/59/60 depending on leap second
zone   = utc-designator / utc-offset
utc-designator = %x5A  ; uppercase "Z"

date          = year    [month  day]
              / year "-" month
              / "--"      month [day]
              / "--"       "-"   day
date-noreduc  = year      month  day
              / "--"       month  day
              / "--"        "-"    day
date-complete = year      month  day

time          = hour [minute [second]] [zone]
              /  "-"  minute [second]  [zone]
              /  "-"   "-"    second   [zone]
time-notrunc  = hour [minute [second]] [zone]
time-complete = hour  minute  second   [zone]
```

```
      time-designator = %x54  ; uppercase "T"
      date-time = date-noreduc  time-designator time-notrunc
      timestamp = date-complete time-designator time-complete

      date-and-or-time = date-time / date / time-designator time

      utc-offset = sign hour [minute]

      Language-Tag = <Language-Tag, defined in [RFC5646], Section 2.1>

      iana-valuespec = <value-spec, see Section 12>
                     ; a publicly defined valuetype format, registered
                     ; with IANA, as defined in Section 12 of this
                     ; document.
```

4.1. TEXT

"text": The "text" value type should be used to identify values that
contain human-readable text. As for the language, it is controlled
by the LANGUAGE property parameter defined in Section 5.1.

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 (U+005C) followed by a Latin
small letter n (U+006E) or a Latin capital letter N (U+004E), that
is, "\n" or "\N".

For example, a multiple line NOTE value of:

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

could be represented as:

```
      NOTE: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.

4.2. URI

   "uri": The "uri" value type should be used to identify values that
   are referenced by a Uniform Resource Identifier (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 Section 3 of [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.example.com/my/picture.jpg
      ldap://ldap.example.com/cn=babs%20jensen

4.3. DATE, TIME, DATE-TIME, DATE-AND-OR-TIME, and TIMESTAMP

   "date", "time", "date-time", "date-and-or-time", and "timestamp":
   Each of these value types is based on the definitions in
   [ISO.8601.2004]. Multiple such values can be specified using the
   comma-separated notation.

   Only the basic format is supported.

4.3.1. DATE

   A calendar date as specified in [ISO.8601.2004], Section 4.1.2.

   Reduced accuracy, as specified in [ISO.8601.2004], Sections 4.1.2.3
   a) and b), but not c), is permitted.

   Expanded representation, as specified in [ISO.8601.2004], Section
   4.1.4, is forbidden.

   Truncated representation, as specified in [ISO.8601.2000], Sections
   5.2.1.3 d), e), and f), is permitted.

   Examples for "date":

            19850412
            1985-04
            1985
            --0412
            ---12

Note the use of YYYY-MM in the second example above. YYYYMM is
disallowed to prevent confusion with YYMMDD. Note also that
YYYY-MM-DD is disallowed since we are using the basic format instead
of the extended format.

4.3.2. TIME

A time of day as specified in [ISO.8601.2004], Section 4.2.

Reduced accuracy, as specified in [ISO.8601.2004], Section 4.2.2.3,
is permitted.

Representation with decimal fraction, as specified in
[ISO.8601.2004], Section 4.2.2.4, is forbidden.

The midnight hour is always represented by 00, never 24 (see
[ISO.8601.2004], Section 4.2.3).

Truncated representation, as specified in [ISO.8601.2000], Sections
5.3.1.4 a), b), and c), is permitted.

Examples for "time":

```
        102200
        1022
        10
        -2200
        --00
        102200Z
        102200-0800
```

4.3.3. DATE-TIME

A date and time of day combination as specified in [ISO.8601.2004],
Section 4.3.

Truncation of the date part, as specified in [ISO.8601.2000], Section
5.4.2 c), is permitted.

Examples for "date-time":

```
        19961022T140000
        --1022T1400
        ---22T14
```

4.3.4. DATE-AND-OR-TIME

Either a DATE-TIME, a DATE, or a TIME value. To allow unambiguous
interpretation, a stand-alone TIME value is always preceded by a "T".

Examples for "date-and-or-time":

```
19961022T140000
--1022T1400
---22T14
19850412
1985-04
1985
--0412
---12
T102200
T1022
T10
T-2200
T--00
T102200Z
T102200-0800
```

4.3.5. TIMESTAMP

A complete date and time of day combination as specified in
[ISO.8601.2004], Section 4.3.2.

Examples for "timestamp":

```
19961022T140000
19961022T140000Z
19961022T140000-05
19961022T140000-0500
```

4.4. BOOLEAN

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

Examples:

```
TRUE
false
True
```

4.5. INTEGER

"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. The maximum value is
9223372036854775807, and the minimum value is -9223372036854775808.
These limits correspond to a signed 64-bit integer using two's-
complement arithmetic.

Examples:

     1234567890
     -1234556790
     +1234556790,432109876

4.6. FLOAT

"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.
Implementations MUST support a precision equal or better than that of
the IEEE "binary64" format [IEEE.754.2008].

   Note: Scientific notation is disallowed. Implementers wishing to
   use their favorite language's %f formatting should be careful.

Examples:

     20.30
     1000000.0000001
     1.333,3.14

4.7. UTC-OFFSET

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

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., +hhmm). 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 basic format for ISO 8601 UTC offsets
MUST be used.

4.8. LANGUAGE-TAG

"language-tag": A single language tag, as defined in [RFC5646].

5. Property Parameters

A property can have attributes associated with it. These "property
parameters" contain meta-information about the property or the
property value. In some cases, the property parameter can be multi-
valued in which case the property parameter value elements are
separated by a COMMA (U+002C).

Property parameter value elements that contain the COLON (U+003A),
SEMICOLON (U+003B), or COMMA (U+002C) character separators MUST be
specified as quoted-string text values. Property parameter values
MUST NOT contain the DQUOTE (U+0022) character. The DQUOTE character
is used as a delimiter for parameter values that contain restricted
characters or URI text.

Applications MUST ignore x-param and iana-param values they don't
recognize.

5.1. LANGUAGE

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 [RFC3282]. The value of the LANGUAGE property parameter is a
language tag as defined in Section 2 of [RFC5646].

Examples:

  ROLE;LANGUAGE=tr:hoca

ABNF:

        language-param = "LANGUAGE=" Language-Tag
          ; Language-Tag is defined in section 2.1 of RFC 5646

5.2. VALUE

The VALUE parameter is OPTIONAL, 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 explicitly
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.

ABNF:

```
   value-param = "VALUE=" value-type

   value-type = "text"
              / "uri"
              / "date"
              / "time"
              / "date-time"
              / "date-and-or-time"
              / "timestamp"
              / "boolean"
              / "integer"
              / "float"
              / "utc-offset"
              / "language-tag"
              / iana-token  ; registered as described in section 12
              / x-name
```

5.3. PREF

The PREF parameter is OPTIONAL and is used to indicate that the
corresponding instance of a property is preferred by the vCard
author. Its value MUST be an integer between 1 and 100 that
quantifies the level of preference. Lower values correspond to a
higher level of preference, with 1 being most preferred.

When the parameter is absent, the default MUST be to interpret the
property instance as being least preferred.

Note that the value of this parameter is to be interpreted only in
relation to values assigned to other instances of the same property
in the same vCard. A given value, or the absence of a value, MUST
NOT be interpreted on its own.

This parameter MAY be applied to any property that allows multiple
instances.

ABNF:

```
        pref-param = "PREF=" (1*2DIGIT / "100")
                            ; An integer between 1 and 100.
```

5.4. ALTID

The ALTID parameter is used to "tag" property instances as being
alternative representations of the same logical property. For
example, translations of a property in multiple languages generates
multiple property instances having different LANGUAGE (Section 5.1)
parameter that are tagged with the same ALTID value.

This parameter's value is treated as an opaque string. Its sole
purpose is to be compared for equality against other ALTID parameter
values.

Two property instances are considered alternative representations of
the same logical property if and only if their names as well as the
value of their ALTID parameters are identical. Property instances
without the ALTID parameter MUST NOT be considered an alternative
representation of any other property instance. Values for the ALTID
parameter are not globally unique: they MAY be reused for different
property names.

Property instances having the same ALTID parameter value count as 1
toward cardinality. Therefore, since N (Section 6.2.2) has
cardinality *1 and TITLE (Section 6.6.1) has cardinality *, these
three examples would be legal:

    N;ALTID=1;LANGUAGE=jp:<U+5C71><U+7530>;<U+592A><U+90CE>;;;
    N;ALTID=1;LANGUAGE=en:Yamada;Taro;;;
    (<U+XXXX> denotes a UTF8-encoded Unicode character.)

    TITLE;ALTID=1;LANGUAGE=fr:Patron
    TITLE;ALTID=1;LANGUAGE=en:Boss

    TITLE;ALTID=1;LANGUAGE=fr:Patron
    TITLE;ALTID=1;LANGUAGE=en:Boss
    TITLE;ALTID=2;LANGUAGE=en:Chief vCard Evangelist

while this one would not:

    N;ALTID=1;LANGUAGE=jp:<U+5C71><U+7530>;<U+592A><U+90CE>;;;
    N:Yamada;Taro;;;
    (Two instances of the N property.)

and these three would be legal but questionable:

    TITLE;ALTID=1;LANGUAGE=fr:Patron
    TITLE;ALTID=2;LANGUAGE=en:Boss
    (Should probably have the same ALTID value.)

```
   TITLE;ALTID=1;LANGUAGE=fr:Patron
   TITLE:LANGUAGE=en:Boss
   (Second line should probably have ALTID=1.)

   N;ALTID=1;LANGUAGE=jp:<U+5C71><U+7530>;<U+592A><U+90CE>;;;
   N;ALTID=1;LANGUAGE=en:Yamada;Taro;;;
   N;ALTID=1;LANGUAGE=en:Smith;John;;;
   (The last line should probably have ALTID=2.  But that would be
    illegal because N has cardinality *1.)
```

   The ALTID property MAY also be used in may contexts other than with
   the LANGUAGE parameter. Here's an example with two representations
   of the same photo in different file formats:

```
   PHOTO;ALTID=1:data:image/jpeg;base64,...
   PHOTO;ALTID=1;data:image/jp2;base64,...
```

   ABNF:

```
        altid-param = "ALTID=" param-value
```

5.5. PID

   The PID parameter is used to identify a specific property among
   multiple instances. It plays a role analogous to the UID property
   (Section 6.7.6) on a per-property instead of per-vCard basis. It MAY
   appear more than once in a given property. It MUST NOT appear on
   properties that may have only one instance per vCard. Its value is
   either a single small positive integer or a pair of small positive
   integers separated by a dot. Multiple values may be encoded in a
   single PID parameter by separating the values with a comma ",". See
   Section 7 for more details on its usage.

   ABNF:

```
        pid-param = "PID=" pid-value *("," pid-value)
        pid-value = 1*DIGIT ["." 1*DIGIT]
```

5.6. TYPE

   The TYPE parameter has multiple, different uses. In general, it is a
   way of specifying class characteristics of the associated property.
   Most of the time, its value is a comma-separated subset of a
   predefined enumeration. In this document, the following properties
   make use of this parameter: FN, NICKNAME, PHOTO, ADR, TEL, EMAIL,
   IMPP, LANG, TZ, GEO, TITLE, ROLE, LOGO, ORG, RELATED, CATEGORIES,

NOTE, SOUND, URL, KEY, FBURL, CALADRURI, and CALURI. The TYPE
parameter MUST NOT be applied on other properties defined in this
document.

The "work" and "home" values act like tags. The "work" value implies
that the property is related to an individual's work place, while the
"home" value implies that the property is related to an individual's
personal life. When neither "work" nor "home" is present, it is
implied that the property is related to both an individual's work
place and personal life in the case that the KIND property's value is
"individual", or to none in other cases.

ABNF:

        type-param = "TYPE=" type-value *("," type-value)

        type-value = "work" / "home" / type-param-tel
                   / type-param-related / iana-token / x-name
          ; This is further defined in individual property sections.

5.7. MEDIATYPE

The MEDIATYPE parameter is used with properties whose value is a URI.
Its use is OPTIONAL. It provides a hint to the vCard consumer
application about the media type [RFC2046] of the resource identified
by the URI. Some URI schemes do not need this parameter. For
example, the "data" scheme allows the media type to be explicitly
indicated as part of the URI [RFC2397]. Another scheme, "http",
provides the media type as part of the URI resolution process, with
the Content-Type HTTP header [RFC2616]. The MEDIATYPE parameter is
intended to be used with URI schemes that do not provide such
functionality (e.g., "ftp" [RFC1738]).

ABNF:

  mediatype-param = "MEDIATYPE=" mediatype
  mediatype = type-name "/" subtype-name *(";" attribute "=" value)
    ; "attribute" and "value" are from [RFC2045]
    ; "type-name" and "subtype-name" are from [RFC4288]

5.8. CALSCALE

The CALSCALE parameter is identical to the CALSCALE property in
iCalendar (see [RFC5545], Section 3.7.1). It is used to define the
calendar system in which a date or date-time value is expressed. The
only value specified by iCalendar is "gregorian", which stands for
the Gregorian system. It is the default when the parameter is
absent. Additional values may be defined in extension documents and

registered with IANA (see Section 10.3.4). A vCard implementation
MUST ignore properties with a CALSCALE parameter value that it does
not understand.

ABNF:

```
calscale-param = "CALSCALE=" calscale-value

calscale-value = "gregorian" / iana-token / x-name
```

5.9. SORT-AS

The "sort-as" parameter is used to specify the string to be used for
national-language-specific sorting. 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 the given strings are used for sorting the vCard.

This parameter's value is a comma-separated list that MUST have as
many or fewer elements as the corresponding property value has
components. This parameter's value is case-sensitive.

ABNF:

```
sort-as-param = "SORT-AS=" sort-as-value

sort-as-value = param-value *("," param-value)
```

Examples: For the case of surname and given name sorting, the
following examples define common sort string usage with the N
property.

```
FN:Rene van der Harten
N;SORT-AS="Harten,Rene":van der Harten;Rene,J.;Sir;R.D.O.N.

FN:Robert Pau Shou Chang
N;SORT-AS="Pau Shou Chang,Robert":Shou Chang;Robert,Pau;;

FN:Osamu Koura
N;SORT-AS="Koura,Osamu":Koura;Osamu;;

FN:Oscar del Pozo
N;SORT-AS="Pozo,Oscar":del Pozo Triscon;Oscar;;

FN:Chistine d'Aboville
N;SORT-AS="Aboville,Christine":d'Aboville;Christine;;
```

```
            FN:H. James de Mann
            N;SORT-AS="Mann,James":de Mann;Henry,James;;
```

   If sorted by surname, the results would be:

```
            Christine d'Aboville
            Rene van der Harten
            Osamu Koura
            H. James de Mann
            Robert Pau Shou Chang
            Oscar del Pozo
```

   If sorted by given name, the results would be:

```
            Christine d'Aboville
            H. James de Mann
            Osamu Koura
            Oscar del Pozo
            Rene van der Harten
            Robert Pau Shou Chang
```

5.10. GEO

   The GEO parameter can be used to indicate global positioning
   information that is specific to an address. Its value is the same as
   that of the GEO property (see Section 6.5.2).

   ABNF:

```
      geo-parameter = "GEO=" DQUOTE URI DQUOTE
```

5.11. TZ

   The TZ parameter can be used to indicate time zone information that
   is specific to an address. Its value is the same as that of the TZ
   property.

   ABNF:

```
      tz-parameter = "TZ=" (param-value / DQUOTE URI DQUOTE)
```

6. vCard Properties

What follows is an enumeration of the standard vCard properties.

6.1. General Properties

6.1.1. BEGIN

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

Value type: text

Cardinality: 1

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

   The BEGIN property is used in conjunction with the END property to
   delimit an entity containing a related set of properties within a
   text/vcard content-type. This construct can be used instead of
   including multiple vCards as body parts inside of a multipart/
   alternative MIME message. 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.

ABNF:

```
  BEGIN-param = 0" "  ; no parameter allowed
  BEGIN-value = "VCARD"
```

Example:

```
      BEGIN:VCARD
```

6.1.2. END

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

Value type: text

Cardinality: 1

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

      The END property is used in conjunction with the BEGIN property to
      delimit an entity containing a related set of properties within a
      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.

   ABNF:

     END-param = 0" " ; no parameter allowed
     END-value = "VCARD"

   Example:

        END:VCARD

6.1.3. SOURCE

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

   Value type: uri

   Cardinality: *

   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.

   ABNF:

     SOURCE-param = "VALUE=uri" / pid-param / pref-param / altid-param
                  / mediatype-param / any-param
     SOURCE-value = URI

   Examples:

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

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

6.1.4. KIND

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

   Value type: A single text value.

   Cardinality: *1

   Special notes: The value may be one of the following:

      "individual" for a vCard representing a single person or entity.
         This is the default kind of vCard.

      "group" for a vCard representing a group of persons or entities.
         The group's member entities can be other vCards or other types
         of entities, such as email addresses or web sites. A group
         vCard will usually contain MEMBER properties to specify the
         members of the group, but it is not required to. A group vCard
         without MEMBER properties can be considered an abstract
         grouping, or one whose members are known empirically (perhaps
         "IETF Participants" or "Republican U.S. Senators").

         All properties in a group vCard apply to the group as a whole,
         and not to any particular MEMBER. For example, an EMAIL
         property might specify the address of a mailing list associated
         with the group, and an IMPP property might refer to a group
         chat room.

      "org" for a vCard representing an organization. An organization
         vCard will not (in fact, MUST NOT) contain MEMBER properties,
         and so these are something of a cross between "individual" and
         "group". An organization is a single entity, but not a person.
         It might represent a business or government, a department or
         division within a business or government, a club, an
         association, or the like.

         All properties in an organization vCard apply to the
         organization as a whole, as is the case with a group vCard.
         For example, an EMAIL property might specify the address of a
         contact point for the organization.

"location" for a named geographical place. A location vCard will
   usually contain a GEO property, but it is not required to. A
   location vCard without a GEO property can be considered an
   abstract location, or one whose definition is known empirically
   (perhaps "New England" or "The Seashore").

   All properties in a location vCard apply to the location
   itself, and not with any entity that might exist at that
   location. For example, in a vCard for an office building, an
   ADR property might give the mailing address for the building,
   and a TEL property might specify the telephone number of the
   receptionist.

An x-name. vCards MAY include private or experimental values for
   KIND. Remember that x-name values are not intended for general
   use and are unlikely to interoperate.

An iana-token. Additional values may be registered with IANA (see
   Section 10.3.4). A new value's specification document MUST
   specify which properties make sense for that new kind of vCard
   and which do not.

Implementations MUST support the specific string values defined
above. If this property is absent, "individual" MUST be assumed
as the default. If this property is present but the
implementation does not understand its value (the value is an
x-name or iana-token that the implementation does not support),
the implementation SHOULD act in a neutral way, which usually
means treating the vCard as though its kind were "individual".
The presence of MEMBER properties MAY, however, be taken as an
indication that the unknown kind is an extension of "group".

Clients often need to visually distinguish contacts based on what
they represent, and the KIND property provides a direct way for
them to do so. For example, when displaying contacts in a list,
an icon could be displayed next to each one, using distinctive
icons for the different kinds; a client might use an outline of a
single person to represent an "individual", an outline of multiple
people to represent a "group", and so on. Alternatively, or in
addition, a client might choose to segregate different kinds of
vCards to different panes, tabs, or selections in the user
interface.

Some clients might also make functional distinctions among the
kinds, ignoring "location" vCards for some purposes and
considering only "location" vCards for others.

When designing those sorts of visual and functional distinctions,
client implementations have to decide how to fit unsupported kinds
into the scheme. What icon is used for them? The one for
"individual"? A unique one, such as an icon of a question mark?
Which tab do they go into? It is beyond the scope of this
specification to answer these questions, but these are things
implementers need to consider.

ABNF:

```
KIND-param = "VALUE=text" / any-param
KIND-value = "individual" / "group" / "org" / "location"
           / iana-token / x-name
```

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
```

6.1.5. XML

Purpose: To include extended XML-encoded vCard data in a plain
   vCard.

Value type: A single text value.

Cardinality: *

Special notes: The content of this property is a single XML 1.0
   [W3C.REC-xml-20081126] element whose namespace MUST be explicitly
   specified using the xmlns attribute and MUST NOT be the vCard 4

      namespace ("urn:ietf:params:xml:ns:vcard-4.0"). (This implies
      that it cannot duplicate a standard vCard property.) The element
      is to be interpreted as if it was contained in a <vcard> element,
      as defined in [RFC6351].

      The fragment is subject to normal line folding and escaping, i.e.,
      replace all backslashes with "\\", then replace all newlines with
      "\n", then fold long lines.

      Support for this property is OPTIONAL, but implementations of this
      specification MUST preserve instances of this property when
      propagating vCards.

      See [RFC6351] for more information on the intended use of this
      property.

   ABNF:

     XML-param = "VALUE=text" / altid-param
     XML-value = text

6.2. Identification Properties

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

6.2.1. FN

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

   Value type: A single text value.

   Cardinality: 1*

   Special notes: This property is based on the semantics of the X.520
      Common Name attribute [CCITT.X520.1988]. The property MUST be
      present in the vCard object.

   ABNF:

     FN-param = "VALUE=text" / type-param / language-param / altid-param
              / pid-param / pref-param / any-param
     FN-value = text

   Example:

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

6.2.2. 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.

Cardinality: *1

Special note: The structured property value corresponds, in
   sequence, to the Family Names (also known as surnames), Given
   Names, Additional Names, Honorific Prefixes, and Honorific
   Suffixes. The text components are separated by the SEMICOLON
   character (U+003B). Individual text components can include
   multiple text values separated by the COMMA character (U+002C).
   This property is based on the semantics of the X.520 individual
   name attributes [CCITT.X520.1988]. The property SHOULD be present
   in the vCard object when the name of the object the vCard
   represents follows the X.520 model.

   The SORT-AS parameter MAY be applied to this property.

ABNF:

```
  N-param = "VALUE=text" / sort-as-param / language-param
          / altid-param / any-param
  N-value = list-component 4(";" list-component)
```

Examples:

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

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

6.2.3. 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
   (U+002C).

Cardinality: *

   Special note: The nickname is the descriptive name given instead of
      or in addition to the one belonging to the object the vCard
      represents. It can also be used to specify a familiar form of a
      proper name specified by the FN or N properties.

   ABNF:

     NICKNAME-param = "VALUE=text" / type-param / language-param
                    / altid-param / pid-param / pref-param / any-param
     NICKNAME-value = text-list

   Examples:

             NICKNAME:Robbie

             NICKNAME:Jim,Jimmie

             NICKNAME;TYPE=work:Boss

6.2.4. PHOTO

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

   Value type: A single URI.

   Cardinality: *

   ABNF:

     PHOTO-param = "VALUE=uri" / altid-param / type-param
                 / mediatype-param / pref-param / pid-param / any-param
     PHOTO-value = URI

   Examples:

       PHOTO:http://www.example.com/pub/photos/jqpublic.gif

       PHOTO:data:image/jpeg;base64,MIICajCCAdOgAwIBAgICBEUwDQYJKoZIhv
        AQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bm
        ljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbiBTeXN0
        <...remainder of base64-encoded data...>

6.2.5. BDAY

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

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

Cardinality: *1

ABNF:

```
  BDAY-param = BDAY-param-date / BDAY-param-text
  BDAY-value = date-and-or-time / text
    ; Value and parameter MUST match.

  BDAY-param-date = "VALUE=date-and-or-time"
  BDAY-param-text = "VALUE=text" / language-param

  BDAY-param =/ altid-param / calscale-param / any-param
    ; calscale-param can only be present when BDAY-value is
    ; date-and-or-time and actually contains a date or date-time.
```

Examples:

```
        BDAY:19960415
        BDAY:--0415
        BDAY;19531015T231000Z
        BDAY;VALUE=text:circa 1800
```

6.2.6. ANNIVERSARY

Purpose: The date of marriage, or equivalent, of the object the
   vCard represents.

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

Cardinality: *1

ABNF:

```
  ANNIVERSARY-param = "VALUE=" ("date-and-or-time" / "text")
  ANNIVERSARY-value = date-and-or-time / text
    ; Value and parameter MUST match.

  ANNIVERSARY-param =/ altid-param / calscale-param / any-param
    ; calscale-param can only be present when ANNIVERSARY-value is
    ; date-and-or-time and actually contains a date or date-time.
```

Examples:

```
        ANNIVERSARY:19960415
```

6.2.7. GENDER

Purpose: To specify the components of the sex and gender identity of
   the object the vCard represents.

Value type: A single structured value with two components. Each
   component has a single text value.

Cardinality: *1

Special notes: The components correspond, in sequence, to the sex
   (biological), and gender identity. Each component is optional.

   Sex component: A single letter. M stands for "male", F stands
      for "female", O stands for "other", N stands for "none or not
      applicable", U stands for "unknown".

   Gender identity component: Free-form text.

ABNF:

                  GENDER-param = "VALUE=text" / any-param
                  GENDER-value = sex [";" text]

                  sex = "" / "M" / "F" / "O" / "N" / "U"

Examples:

   GENDER:M
   GENDER:F
   GENDER:M;Fellow
   GENDER:F;grrrl
   GENDER:O;intersex
   GENDER:;it's complicated

6.3. Delivery Addressing Properties

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

6.3.1. ADR

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

Value type: A single structured text value, separated by the
   SEMICOLON character (U+003B).

Cardinality: *

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 (full name in the language specified in
      Section 5.1).

   When a component value is missing, the associated component
   separator MUST still be specified.

   Experience with vCard 3 has shown that the first two components
   (post office box and extended address) are plagued with many
   interoperability issues. To ensure maximal interoperability,
   their values SHOULD be empty.

   The text components are separated by the SEMICOLON character
   (U+003B). 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
   (U+002C).

   The property can include the "PREF" parameter to indicate the
   preferred delivery address when more than one address is
   specified.

   The GEO and TZ parameters MAY be used with this property.

   The property can also include a "LABEL" parameter to present a
   delivery address label for the address. Its value is a plain-text
   string representing the formatted address. Newlines are encoded
   as \n, as they are for property values.

ABNF:

   label-param = "LABEL=" param-value

   ADR-param = "VALUE=text" / label-param / language-param
             / geo-parameter / tz-parameter / altid-param / pid-param
             / pref-param / type-param / any-param

```
ADR-value = ADR-component-pobox ";" ADR-component-ext ";"
            ADR-component-street ";" ADR-component-locality ";"
            ADR-component-region ";" ADR-component-code ";"
            ADR-component-country
ADR-component-pobox    = list-component
ADR-component-ext      = list-component
ADR-component-street   = list-component
ADR-component-locality = list-component
ADR-component-region   = list-component
ADR-component-code     = list-component
ADR-component-country  = list-component
```

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

```
ADR;GEO="geo:12.3457,78.910";LABEL="Mr. John Q. Public, Esq.\n
 Mail Drop: TNE QB\n123 Main Street\nAny Town, CA  91921-1234\n
 U.S.A.":;;123 Main Street;Any Town;CA;91921-1234;U.S.A.
```

6.4. Communications Properties

These properties describe information about how to communicate with
the object the vCard represents.

6.4.1. TEL

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

Value type: By default, it is a single free-form text value (for
   backward compatibility with vCard 3), but it SHOULD be reset to a
   URI value. It is expected that the URI scheme will be "tel", as
   specified in [RFC3966], but other schemes MAY be used.

Cardinality: *

Special notes: This property is based on the X.520 Telephone Number
   attribute [CCITT.X520.1988].

   The property can include the "PREF" parameter to indicate a
   preferred-use telephone number.

   The property can include the parameter "TYPE" to specify intended
   use for the telephone number. The predefined values for the TYPE
   parameter are:

```
+-----------+--------------------------------------------------------+
| Value     | Description                                            |
+-----------+--------------------------------------------------------+
| text      | Indicates that the telephone number supports text      |
|           | messages (SMS).                                        |
| voice     | Indicates a voice telephone number.                    |
| fax       | Indicates a facsimile telephone number.                |
| cell      | Indicates a cellular or mobile telephone number.       |
| video     | Indicates a video conferencing telephone number.       |
| pager     | Indicates a paging device telephone number.            |
| textphone | Indicates a telecommunication device for people with   |
|           | hearing or speech difficulties.                        |
+-----------+--------------------------------------------------------+
```

The default type is "voice". These type parameter values can be
specified as a parameter list (e.g., TYPE=text;TYPE=voice) or as a
value list (e.g., TYPE="text,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 VOICE and FAX telephone number by the value list
TYPE="voice,fax".

If this property's value is a URI that can also be used for
instant messaging, the IMPP (Section 6.4.3) property SHOULD be
used in addition to this property.

ABNF:

```
TEL-param = TEL-text-param / TEL-uri-param
TEL-value = TEL-text-value / TEL-uri-value
  ; Value and parameter MUST match.

TEL-text-param = "VALUE=text"
TEL-text-value = text

TEL-uri-param = "VALUE=uri" / mediatype-param
TEL-uri-value = URI

TEL-param =/ type-param / pid-param / pref-param / altid-param
           / any-param

type-param-tel = "text" / "voice" / "fax" / "cell" / "video"
               / "pager" / "textphone" / iana-token / x-name
  ; type-param-tel MUST NOT be used with a property other than TEL.
```

   Example:

     TEL;VALUE=uri;PREF=1;TYPE="voice,home":tel:+1-555-555-5555;ext=5555
     TEL;VALUE=uri;TYPE=home:tel:+33-01-23-45-67

6.4.2. EMAIL

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

   Value type: A single text value.

   Cardinality: *

   Special notes: The property can include tye "PREF" parameter to
      indicate a preferred-use email address when more than one is
      specified.

      Even though the value is free-form UTF-8 text, it is likely to be
      interpreted by a Mail User Agent (MUA) as an "addr-spec", as
      defined in [RFC5322], Section 3.4.1. Readers should also be aware
      of the current work toward internationalized email addresses
      [RFC5335bis].

   ABNF:

     EMAIL-param = "VALUE=text" / pid-param / pref-param / type-param
                 / altid-param / any-param
     EMAIL-value = text

   Example:

           EMAIL;TYPE=work:jqpublic@xyz.example.com

           EMAIL;PREF=1:jane_doe@example.com

6.4.3. IMPP

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

   Value type: A single URI.

   Cardinality: *

   Special notes: The property may include the "PREF" parameter to
      indicate that this is a preferred address and has the same
      semantics as the "PREF" parameter in a TEL property.

If this property's value is a URI that can be used for voice
and/or video, the TEL property (Section 6.4.1) SHOULD be used in
addition to this property.

This property is adapted from [RFC4770], which is made obsolete by
this document.

ABNF:

```
IMPP-param = "VALUE=uri" / pid-param / pref-param / type-param
             / mediatype-param / altid-param / any-param
IMPP-value = URI
```

Example:

```
IMPP;PREF=1:xmpp:alice@example.com
```

6.4.4. LANG

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

Value type: A single language-tag value.

Cardinality: *

ABNF:

```
LANG-param = "VALUE=language-tag" / pid-param / pref-param
             / altid-param / type-param / any-param
LANG-value = Language-Tag
```

Example:

```
LANG;TYPE=work;PREF=1:en
LANG;TYPE=work;PREF=2:fr
LANG;TYPE=home:fr
```

6.5. Geographical Properties

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

6.5.1. TZ

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

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

   Cardinality: *

   Special notes: It is expected that names from the public-domain
      Olson database [TZ-DB] will be used, but this is not a
      restriction. See also [IANA-TZ].

      Efforts are currently being directed at creating a standard URI
      scheme for expressing time zone information. Usage of such a
      scheme would ensure a high level of interoperability between
      implementations that support it.

      Note that utc-offset values SHOULD NOT be used because the UTC
      offset varies with time -- not just because of the usual daylight
      saving time shifts that occur in may regions, but often entire
      regions will "re-base" their overall offset. The actual offset
      may be +/- 1 hour (or perhaps a little more) than the one given.

   ABNF:

     TZ-param = "VALUE=" ("text" / "uri" / "utc-offset")
     TZ-value = text / URI / utc-offset
       ; Value and parameter MUST match.

     TZ-param =/ altid-param / pid-param / pref-param / type-param
                / mediatype-param / any-param

   Examples:

     TZ:Raleigh/North America

     TZ;VALUE=utc-offset:-0500
       ; Note: utc-offset format is NOT RECOMMENDED.

6.5.2. GEO

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

   Value type: A single URI.

   Cardinality: *

   Special notes: The "geo" URI scheme [RFC5870] is particularly well
      suited for this property, but other schemes MAY be used.

ABNF:

```
  GEO-param = "VALUE=uri" / pid-param / pref-param / type-param
            / mediatype-param / altid-param / any-param
  GEO-value = URI
```

Example:

```
      GEO:geo:37.386013,-122.082932
```

6.6. Organizational Properties

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

6.6.1. TITLE

Purpose: To specify the position or job of the object the vCard
   represents.

Value type: A single text value.

Cardinality: *

Special notes: This property is based on the X.520 Title attribute
   [CCITT.X520.1988].

ABNF:

```
  TITLE-param = "VALUE=text" / language-param / pid-param
             / pref-param / altid-param / type-param / any-param
  TITLE-value = text
```

Example:

```
      TITLE:Research Scientist
```

6.6.2. ROLE

Purpose: To specify the function or part played in a particular
   situation by the object the vCard represents.

Value type: A single text value.

Cardinality: *

Special notes: This property is based on the X.520 Business Category
   explanatory attribute [CCITT.X520.1988]. 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.

ABNF:

     ROLE-param = "VALUE=text" / language-param / pid-param / pref-param
                  / type-param / altid-param / any-param
     ROLE-value = text

Example:

          ROLE:Project Leader

6.6.3. LOGO

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

Value type: A single URI.

Cardinality: *

ABNF:

     LOGO-param = "VALUE=uri" / language-param / pid-param / pref-param
                  / type-param / mediatype-param / altid-param / any-param
     LOGO-value = URI

Examples:

     LOGO:http://www.example.com/pub/logos/abccorp.jpg

     LOGO:data:image/jpeg;base64,MIICajCCAdOgAwIBAgICBEUwDQYJKoZIhvc
      AQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bm
      ljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbiBTeXN0
      <...the remainder of base64-encoded data...>

6.6.4. ORG

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

Value type: A single structured text value consisting of components
   separated by the SEMICOLON character (U+003B).

    Cardinality: *

    Special notes: The property is based on the X.520 Organization Name
       and Organization Unit attributes [CCITT.X520.1988]. The property
       value is a structured type consisting of the organization name,
       followed by zero or more levels of organizational unit names.

       The SORT-AS parameter MAY be applied to this property.

    ABNF:

      ORG-param = "VALUE=text" / sort-as-param / language-param
                / pid-param / pref-param / altid-param / type-param
                / any-param
      ORG-value = component *(";" component)

    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

6.6.5. MEMBER

    Purpose: To include a member in the group this vCard represents.

    Value type: A single URI. It MAY refer to something other than a
       vCard object. For example, an email distribution list could
       employ the "mailto" URI scheme [RFC6068] for efficiency.

    Cardinality: *

    Special notes: This property MUST NOT be present unless the value of
       the KIND property is "group".

    ABNF:

      MEMBER-param = "VALUE=uri" / pid-param / pref-param / altid-param
                   / mediatype-param / any-param
      MEMBER-value = URI

Examples:

```
BEGIN:VCARD
VERSION:4.0
KIND:group
FN:The Doe family
MEMBER:urn:uuid:03a0e51f-d1aa-4385-8a53-e29025acd8af
MEMBER:urn:uuid:b8767877-b4a1-4c70-9acc-505d3819e519
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:John Doe
UID:urn:uuid:03a0e51f-d1aa-4385-8a53-e29025acd8af
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:Jane Doe
UID:urn:uuid:b8767877-b4a1-4c70-9acc-505d3819e519
END:VCARD

BEGIN:VCARD
VERSION:4.0
KIND:group
FN:Funky distribution list
MEMBER:mailto:subscriber1@example.com
MEMBER:xmpp:subscriber2@example.com
MEMBER:sip:subscriber3@example.com
MEMBER:tel:+1-418-555-5555
END:VCARD
```

6.6.6. RELATED

Purpose: To specify a relationship between another entity and the
   entity represented by this vCard.

Value type: A single URI. It can also be reset to a single text
   value. The text value can be used to specify textual information.

Cardinality: *

Special notes: The TYPE parameter MAY be used to characterize the
   related entity. It contains a comma-separated list of values that
   are registered with IANA as described in Section 10.2. The
   registry is pre-populated with the values defined in [xfn]. This
   document also specifies two additional values:

   agent: an entity who may sometimes act on behalf of the entity
      associated with the vCard.

      emergency: indicates an emergency contact

   ABNF:

     RELATED-param = RELATED-param-uri / RELATED-param-text
     RELATED-value = URI / text
       ; Parameter and value MUST match.

     RELATED-param-uri = "VALUE=uri" / mediatype-param
     RELATED-param-text = "VALUE=text" / language-param

     RELATED-param =/ pid-param / pref-param / altid-param / type-param
                    / any-param

     type-param-related = related-type-value *("," related-type-value)
       ; type-param-related MUST NOT be used with a property other than
       ; RELATED.

     related-type-value = "contact" / "acquaintance" / "friend" / "met"
                       / "co-worker" / "colleague" / "co-resident"
                       / "neighbor" / "child" / "parent"
                       / "sibling" / "spouse" / "kin" / "muse"
                       / "crush" / "date" / "sweetheart" / "me"
                       / "agent" / "emergency"

   Examples:

   RELATED;TYPE=friend:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
   RELATED;TYPE=contact:http://example.com/directory/jdoe.vcf
   RELATED;TYPE=co-worker;VALUE=text:Please contact my assistant Jane
    Doe for any inquiries.

6.7. Explanatory Properties

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

6.7.1. CATEGORIES

   Purpose: To specify application category information about the
      vCard, also known as "tags".

   Value type: One or more text values separated by a COMMA character
      (U+002C).

   Cardinality: *

ABNF:

```
   CATEGORIES-param = "VALUE=text" / pid-param / pref-param
                    / type-param / altid-param / any-param
   CATEGORIES-value = text-list
```

Example:

```
        CATEGORIES:TRAVEL AGENT

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

6.7.2. NOTE

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

Value type: A single text value.

Cardinality: *

Special notes: The property is based on the X.520 Description
   attribute [CCITT.X520.1988].

ABNF:

```
   NOTE-param = "VALUE=text" / language-param / pid-param / pref-param
              / type-param / altid-param / any-param
   NOTE-value = text
```

Example:

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

6.7.3. PRODID

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

Type value: A single text value.

Cardinality: *1

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 ensure that the text
   value is unique.

ABNF:

```
  PRODID-param = "VALUE=text" / any-param
  PRODID-value = text
```

Example:

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

6.7.4. REV

Purpose: To specify revision information about the current vCard.

Value type: A single timestamp value.

Cardinality: *1

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

ABNF:

```
  REV-param = "VALUE=timestamp" / any-param
  REV-value = timestamp
```

Example:

```
      REV:19951031T222710Z
```

6.7.5. SOUND

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

Value type: A single URI.

Cardinality: *

ABNF:

```
  SOUND-param = "VALUE=uri" / language-param / pid-param / pref-param
             / type-param / mediatype-param / altid-param
             / any-param
  SOUND-value = URI
```

   Example:

      SOUND:CID:JOHNQPUBLIC.part8.19960229T080000.xyzMail@example.com

      SOUND:data:audio/basic;base64,MIICajCCAdOgAwIBAgICBEUwDQYJKoZIh
       AQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bm
       ljYXRpb25zIENvcnBvcmF0aW9uMRwwGgYDVQQLExNJbmZvcm1hdGlvbiBTeXN0
       <...the remainder of base64-encoded data...>

6.7.6. UID

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

   Value type: A single URI value. It MAY also be reset to free-form
      text.

   Cardinality: *1

   Special notes: This property is used to uniquely identify the object
      that the vCard represents. The "uuid" URN namespace defined in
      [RFC4122] is particularly well suited to this task, but other URI
      schemes MAY be used. Free-form text MAY also be used.

   ABNF:

     UID-param = UID-uri-param / UID-text-param
     UID-value = UID-uri-value / UID-text-value
       ; Value and parameter MUST match.

     UID-uri-param = "VALUE=uri"
     UID-uri-value = URI

     UID-text-param = "VALUE=text"
     UID-text-value = text

     UID-param =/ any-param

   Example:

         UID:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

6.7.7. CLIENTPIDMAP

Purpose: To give a global meaning to a local PID source identifier.

Value type: A semicolon-separated pair of values. The first field
   is a small integer corresponding to the second field of a PID
   parameter instance. The second field is a URI. The "uuid" URN
   namespace defined in [RFC4122] is particularly well suited to this
   task, but other URI schemes MAY be used.

Cardinality: *

Special notes: PID source identifiers (the source identifier is the
   second field in a PID parameter instance) are small integers that
   only have significance within the scope of a single vCard
   instance. Each distinct source identifier present in a vCard MUST
   have an associated CLIENTPIDMAP. See Section 7 for more details
   on the usage of CLIENTPIDMAP.

   PID source identifiers MUST be strictly positive. Zero is not
   allowed.

   As a special exception, the PID parameter MUST NOT be applied to
   this property.

ABNF:

   CLIENTPIDMAP-param = any-param
   CLIENTPIDMAP-value = 1*DIGIT ";" URI

Example:

   TEL;PID=3.1,4.2;VALUE=uri:tel:+1-555-555-5555
   EMAIL;PID=4.1,5.2:jdoe@example.com
   CLIENTPIDMAP:1;urn:uuid:3df403f4-5924-4bb7-b077-3c711d9eb34b
   CLIENTPIDMAP:2;urn:uuid:d89c9c7a-2e1b-4832-82de-7e992d95faa5

6.7.8. URL

Purpose: To specify a uniform resource locator associated with the
   object to which the vCard refers. Examples for individuals
   include personal web sites, blogs, and social networking site
   identifiers.

Cardinality: *

Value type: A single uri value.

ABNF:

```
   URL-param = "VALUE=uri" / pid-param / pref-param / type-param
               / mediatype-param / altid-param / any-param
   URL-value = URI
```

Example:

```
        URL:http://example.org/restaurant.french/~chezchic.html
```

6.7.9. VERSION

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

Value type: A single text value.

Cardinality: 1

Special notes: This property MUST be present in the vCard object,
   and it must appear immediately after BEGIN:VCARD. The value MUST
   be "4.0" if the vCard corresponds to this specification. Note
   that earlier versions of vCard allowed this property to be placed
   anywhere in the vCard object, or even to be absent.

ABNF:

```
   VERSION-param = "VALUE=text" / any-param
   VERSION-value = "4.0"
```

Example:

```
        VERSION:4.0
```

6.8. Security Properties

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

6.8.1. KEY

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

Value type: A single URI. It can also be reset to a text value.

Cardinality: *

ABNF:

```
KEY-param = KEY-uri-param / KEY-text-param
KEY-value = KEY-uri-value / KEY-text-value
  ; Value and parameter MUST match.

KEY-uri-param = "VALUE=uri" / mediatype-param
KEY-uri-value = URI

KEY-text-param = "VALUE=text"
KEY-text-value = text

KEY-param =/ altid-param / pid-param / pref-param / type-param
            / any-param
```

Examples:

```
KEY:http://www.example.com/keys/jdoe.cer

KEY;MEDIATYPE=application/pgp-keys:ftp://example.com/keys/jdoe

KEY:data:application/pgp-keys;base64,MIICajCCAdOgAwIBAgICBE
 UwDQYJKoZIhvcNAQEEBQAwdzELMAkGA1UEBhMCVVMxLDAqBgNVBAoTI05l
 <... remainder of base64-encoded data ...>
```

6.9. Calendar Properties

These properties are further specified in [RFC2739].

6.9.1. FBURL

Purpose: To specify the URI for the busy time associated with the
   object that the vCard represents.

Value type: A single URI value.

Cardinality: *

Special notes: Where multiple FBURL properties are specified, the
   default FBURL property is indicated with the PREF parameter. The
   FTP [RFC1738] or HTTP [RFC2616] type of URI points to an iCalendar
   [RFC5545] object associated with a snapshot of the next few weeks
   or months of busy time data. If the iCalendar object is
   represented as a file or document, its file extension should be
   ".ifb".

ABNF:

```
FBURL-param = "VALUE=uri" / pid-param / pref-param / type-param
              / mediatype-param / altid-param / any-param
FBURL-value = URI
```

Examples:

```
FBURL;PREF=1:http://www.example.com/busy/janedoe
FBURL;MEDIATYPE=text/calendar:ftp://example.com/busy/project-a.ifb
```

6.9.2. CALADRURI

Purpose: To specify the calendar user address [RFC5545] to which a
   scheduling request [RFC5546] should be sent for the object
   represented by the vCard.

Value type: A single URI value.

Cardinality: *

Special notes: Where multiple CALADRURI properties are specified,
   the default CALADRURI property is indicated with the PREF
   parameter.

ABNF:

```
CALADRURI-param = "VALUE=uri" / pid-param / pref-param / type-param
                  / mediatype-param / altid-param / any-param
CALADRURI-value = URI
```

Example:

```
CALADRURI;PREF=1:mailto:janedoe@example.com
CALADRURI:http://example.com/calendar/jdoe
```

6.9.3. CALURI

Purpose: To specify the URI for a calendar associated with the
   object represented by the vCard.

Value type: A single URI value.

Cardinality: *

Special notes: Where multiple CALURI properties are specified, the
   default CALURI property is indicated with the PREF parameter. The
   property should contain a URI pointing to an iCalendar [RFC5545]

object associated with a snapshot of the user's calendar store.
If the iCalendar object is represented as a file or document, its
file extension should be ".ics".

ABNF:

```
CALURI-param = "VALUE=uri" / pid-param / pref-param / type-param
               / mediatype-param / altid-param / any-param
CALURI-value = URI
```

Examples:

```
CALURI;PREF=1:http://cal.example.com/calA
CALURI;MEDIATYPE=text/calendar:ftp://ftp.example.com/calA.ics
```

6.10. 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.

7. Synchronization

vCard data often needs to be synchronized between devices. In this
context, synchronization is defined as the intelligent merging of two
representations of the same object. vCard 4.0 includes mechanisms to
aid this process.

7.1. Mechanisms

Two mechanisms are available: the UID property is used to match
multiple instances of the same vCard, while the PID parameter is used
to match multiple instances of the same property.

The term "matching" is used here to mean recognizing that two
instances are in fact representations of the same object. For
example, a single vCard that is shared with someone results in two
vCard instances. After they have evolved separately, they still
represent the same object, and therefore may be matched by a
synchronization engine.

7.1.1. Matching vCard Instances

vCard instances for which the UID properties (Section 6.7.6) are
equivalent MUST be matched. Equivalence is determined as specified
in [RFC3986], Section 6.

In all other cases, vCard instances MAY be matched at the discretion
of the synchronization engine.

7.1.2. Matching Property Instances

Property instances belonging to unmatched vCards MUST NOT be matched.

Property instances whose name (e.g., EMAIL, TEL, etc.) is not the
same MUST NOT be matched.

Property instances whose name is CLIENTPIDMAP are handled separately
and MUST NOT be matched. The synchronization MUST ensure that there
is consistency of CLIENTPIDMAPs among matched vCard instances.

Property instances belonging to matched vCards, whose name is the
same, and whose maximum cardinality is 1, MUST be matched.

Property instances belonging to matched vCards, whose name is the
same, and whose PID parameters match, MUST be matched. See
Section 7.1.3 for details on PID matching.

In all other cases, property instances MAY be matched at the
discretion of the synchronization engine.

7.1.3. PID Matching

Two PID values for which the first fields are equivalent represent
the same local value.

Two PID values representing the same local value and for which the
second fields point to CLIENTPIDMAP properties whose second field
URIs are equivalent (as specified in [RFC3986], Section 6) also
represent the same global value.

PID parameters for which at least one pair of their values represent
the same global value MUST be matched.

In all other cases, PID parameters MAY be matched at the discretion
of the synchronization engine.

For example, PID value "5.1", in the first vCard below, and PID value
"5.2", in the second vCard below, represent the same global value.

```
BEGIN:VCARD
VERSION:4.0
EMAIL;PID=4.2,5.1:jdoe@example.com
CLIENTPIDMAP:1;urn:uuid:3eef374e-7179-4196-a914-27358c3e6527
CLIENTPIDMAP:2;urn:uuid:42bcd5a7-1699-4514-87b4-056edf68e9cc
END:VCARD

BEGIN:VCARD
VERSION:4.0
EMAIL;PID=5.1,5.2:john@example.com
CLIENTPIDMAP:1;urn:uuid:0c75c629-6a8d-4d5e-a07f-1bb35846854d
CLIENTPIDMAP:2;urn:uuid:3eef374e-7179-4196-a914-27358c3e6527
END:VCARD
```

7.2. Example

7.2.1. Creation

The following simple vCard is first created on a given device.

```
BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN;PID=1.1:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
END:VCARD
```

This new vCard is assigned the UID
"urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1" by the creating
device. The FN and EMAIL properties are assigned the same local
value of 1, and this value is given global context by associating it
with "urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556", which
represents the creating device. We are at liberty to reuse the same
local value since instances of different properties will never be
matched. The N property has no PID because it is forbidden by its
maximum cardinality of 1.

7.2.2. Initial Sharing

This vCard is shared with a second device. Upon inspecting the UID
property, the second device understands that this is a new vCard
(i.e., unmatched) and thus the synchronization results in a simple
copy.

7.2.3. Adding and Sharing a Property

A new phone number is created on the first device, then the vCard is
shared with the second device. This is what the second device
receives:

```
BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN;PID=1.1:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
TEL;PID=1.1;VALUE=uri:tel:+1-555-555-5555
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
END:VCARD
```

Upon inspecting the UID property, the second device matches the vCard
it received to the vCard that it already has stored. It then starts
comparing the properties of the two vCards in same-named pairs.

The FN properties are matched because the PID parameters have the
same global value. Since the property value is the same, no update
takes place.

The N properties are matched automatically because their maximum
cardinality is 1. Since the property value is the same, no update
takes place.

The EMAIL properties are matched because the PID parameters have the
same global value. Since the property value is the same, no update
takes place.

The TEL property in the new vCard is not matched to any in the stored
vCard because no property in the stored vCard has the same name.
Therefore, this property is copied from the new vCard to the stored
vCard.

The CLIENTPIDMAP property is handled separately by the
synchronization engine. It ensures that it is consistent with the
stored one. If it was not, the results would be up to the
synchronization engine, and thus undefined by this document.

7.2.4. Simultaneous Editing

A new email address and a new phone number are added to the vCard on
each of the two devices, and then a new synchronization event
happens. Here are the vCards that are communicated to each other:

```
BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN;PID=1.1:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
EMAIL;PID=2.1:boss@example.com
TEL;PID=1.1;VALUE=uri:tel:+1-555-555-5555
TEL;PID=2.1;VALUE=uri:tel:+1-666-666-6666
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
END:VCARD

BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN;PID=1.1:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
EMAIL;PID=2.2:ceo@example.com
TEL;PID=1.1;VALUE=uri:tel:+1-555-555-5555
TEL;PID=2.2;VALUE=uri:tel:+1-666-666-6666
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
CLIENTPIDMAP:2;urn:uuid:1f762d2b-03c4-4a83-9a03-75ff658a6eee
END:VCARD
```

On the first device, the same PID source identifier (1) is reused for
the new EMAIL and TEL properties. On the second device, a new source
identifier (2) is generated, and a corresponding CLIENTPIDMAP
property is created. It contains the second device's identifier,
"urn:uuid:1f762d2b-03c4-4a83-9a03-75ff658a6eee".

The new EMAIL properties are unmatched on both sides since the PID
global value is new in both cases. The sync thus results in a copy
on both sides.

Although the situation appears to be the same for the TEL properties,
in this case, the synchronization engine is particularly smart and
matches the two new TEL properties even though their PID global
values are different. Note that in this case, the rules of
Section 7.1.2 state that two properties MAY be matched at the
discretion of the synchronization engine. Therefore, the two
properties are merged.

All this results in the following vCard, which is stored on both
devices:

```
BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
EMAIL;PID=2.1:boss@example.com
EMAIL;PID=2.2:ceo@example.com
TEL;PID=1.1;VALUE=uri:tel:+1-555-555-5555
TEL;PID=2.1,2.2;VALUE=uri:tel:+1-666-666-6666
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
CLIENTPIDMAP:2;urn:uuid:1f762d2b-03c4-4a83-9a03-75ff658a6eee
END:VCARD
```

7.2.5. Global Context Simplification

The two devices finish their synchronization procedure by simplifying
their global contexts. Since they haven't talked to any other
device, the following vCard is for all purposes equivalent to the
above. It is also shorter.

```
BEGIN:VCARD
VERSION:4.0
UID:urn:uuid:4fbe8971-0bc3-424c-9c26-36c3e1eff6b1
FN:J. Doe
N:Doe;J.;;;
EMAIL;PID=1.1:jdoe@example.com
EMAIL;PID=2.1:boss@example.com
EMAIL;PID=3.1:ceo@example.com
TEL;PID=1.1;VALUE=uri:tel:+1-555-555-5555
TEL;PID=2.1;VALUE=uri:tel:+1-666-666-6666
CLIENTPIDMAP:1;urn:uuid:53e374d9-337e-4727-8803-a1e9c14e0556
END:VCARD
```

The details of global context simplification are unspecified by this
document. They are left up to the synchronization engine. This
example is merely intended to illustrate the possibility, which
investigating would be, in the author's opinion, worthwhile.

8. Example: Author's vCard

```
BEGIN:VCARD
VERSION:4.0
FN:Simon Perreault
N:Perreault;Simon;;;ing. jr,M.Sc.
BDAY:--0203
ANNIVERSARY:20090808T1430-0500
GENDER:M
```

```
   LANG;PREF=1:fr
   LANG;PREF=2:en
   ORG;TYPE=work:Viagenie
   ADR;TYPE=work:;Suite D2-630;2875 Laurier;
    Quebec;QC;G1V 2M2;Canada
   TEL;VALUE=uri;TYPE="work,voice";PREF=1:tel:+1-418-656-9254;ext=102
   TEL;VALUE=uri;TYPE="work,cell,voice,video,text":tel:+1-418-262-6501
   EMAIL;TYPE=work:simon.perreault@viagenie.ca
   GEO;TYPE=work:geo:46.772673,-71.282945
   KEY;TYPE=work;VALUE=uri:
    http://www.viagenie.ca/simon.perreault/simon.asc
   TZ:-0500
   URL;TYPE=home:http://nomis80.org
   END:VCARD
```

9. Security Considerations

   o Internet mail is often used to transport vCards and 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 or confidentiality can no longer be
      guaranteed. Applications should also take care to display
      directory data in a "safe" environment.

   o vCards can carry cryptographic keys or certificates, as described
      in Section 6.8.1.

   o vCards often carry information that can be sensitive (e.g.,
      birthday, address, and phone information). Although vCards have
      no inherent authentication or confidentiality provisions, they can
      easily be carried by any security mechanism that transfers MIME
      objects to address authentication or confidentiality (e.g., S/MIME
      [RFC5751], OpenPGP [RFC4880]). In cases where the confidentiality
      or authenticity of information contained in vCard is a concern,
      the vCard SHOULD be transported using one of these secure
      mechanisms. The KEY property (Section 6.8.1) can be used to
      transport the public key used by these mechanisms.

   o 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
      SOURCE property (Section 6.1.3) SHOULD be specified. In addition,
      the "REV" type described in Section 6.7.4 can be specified to
      indicate the last time that the vCard data was updated.

   o Many vCard properties may be used to transport URIs. Please refer
      to [RFC3986], Section 7, for considerations related to URIs.

10. IANA Considerations

10.1. Media Type Registration

IANA has registered the following Media Type (in
<http://www.iana.org/>) and marked the text/directory Media Type as
DEPRECATED.

To: ietf-types@iana.org

Subject: Registration of media type text/vcard

Type name: text

Subtype name: vcard

Required parameters: none

Optional parameters: version

   The "version" parameter is to be interpreted identically as the
   VERSION vCard property. If this parameter is present, all vCards
   in a text/vcard body part MUST have a VERSION property with value
   identical to that of this MIME parameter.

   "charset": as defined for text/plain [RFC2046]; encodings other
   than UTF-8 [RFC3629] MUST NOT be used.

Encoding considerations: 8bit

Security considerations: See Section 9.

Interoperability considerations: The text/vcard media type is
   intended to identify vCard data of any version. There are older
   specifications of vCard [RFC2426][vCard21] 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 6.7.9) 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; profile=vcard
   * text/x-vcard

Published specification: RFC 6350

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

Additional information:

    Magic number(s):

    File extension(s): .vcf .vcard

    Macintosh file type code(s):

Person & email address to contact for further information: vCard
    discussion mailing list <vcarddav@ietf.org>

Intended usage: COMMON

Restrictions on usage: none

Author: Simon Perreault

Change controller: IETF

10.2. Registering New vCard Elements

This section defines the process for registering new or modified
vCard elements (i.e., properties, parameters, value data types, and
values) with IANA.

10.2.1. Registration Procedure

The IETF has created a mailing list, vcarddav@ietf.org, which can be
used for public discussion of vCard element proposals prior to
registration. Use of the mailing list is strongly encouraged. The
IESG has appointed a designated expert who will monitor the
vcarddav@ietf.org mailing list and review registrations.

Registration of new vCard elements MUST be reviewed by the designated
expert and published in an RFC. A Standards Track RFC is REQUIRED
for the registration of new value data types that modify existing
properties. A Standards Track RFC is also REQUIRED for registration
of vCard elements that modify vCard elements previously documented in
a Standards Track RFC.

The registration procedure begins when a completed registration
template, defined in the sections below, is sent to vcarddav@ietf.org
and iana@iana.org. Within two weeks, the designated expert is
expected to tell IANA and the submitter of the registration whether
the registration is approved, approved with minor changes, or
rejected with cause. When a registration is rejected with cause, it
can be re-submitted if the concerns listed in the cause are
addressed. Decisions made by the designated expert can be appealed
to the IESG Applications Area Director, then to the IESG. They
follow the normal appeals procedure for IESG decisions.

Once the registration procedure concludes successfully, IANA creates
or modifies the corresponding record in the vCard registry. The
completed registration template is discarded.

An RFC specifying new vCard elements MUST include the completed
registration templates, which MAY be expanded with additional
information. These completed templates are intended to go in the
body of the document, not in the IANA Considerations section.

Finally, note that there is an XML representation for vCard defined
in [RFC6351]. An XML representation SHOULD be defined for new vCard
elements.

10.2.2. Vendor Namespace

The vendor namespace is used for vCard elements associated with
commercially available products. "Vendor" or "producer" are
construed as equivalent and very broadly in this context.

A registration may be placed in the vendor namespace by anyone who
needs to interchange files associated with the particular product.
However, the registration formally belongs to the vendor or
organization handling the vCard elements in the namespace being
registered. Changes to the specification will be made at their
request, as discussed in subsequent sections.

vCard elements belonging to the vendor namespace will be
distinguished by the "VND-" prefix. This is followed by an IANA-
registered Private Enterprise Number (PEN), a dash, and a vCard
element designation of the vendor's choosing (e.g., "VND-123456-
MUDPIE").

While public exposure and review of vCard elements to be registered
in the vendor namespace are not required, using the vcarddav@ietf.org
mailing list for review is strongly encouraged to improve the quality
of those specifications. Registrations in the vendor namespace may
be submitted directly to the IANA.

10.2.3. Registration Template for Properties

A property is defined by completing the following template.

Namespace: Empty for the global namespace, "VND-NNNN-" for a vendor-
   specific property (where NNNN is replaced by the vendor's PEN).

Property name: The name of the property.

Purpose: The purpose of the property. Give a short but clear
   description.

Value type: Any of the valid value types for the property value
   needs to be specified. The default value type also needs to be
   specified.

Cardinality: See Section 6.

Property parameters: Any of the valid property parameters for the
   property MUST be specified.

Description: Any special notes about the property, how it is to be
   used, etc.

Format definition: The ABNF for the property definition needs to be
   specified.

Example(s): One or more examples of instances of the property need
   to be specified.

10.2.4. Registration Template for Parameters

A parameter is defined by completing the following template.

Namespace: Empty for the global namespace, "VND-NNNN-" for a vendor-
   specific property (where NNNN is replaced by the vendor's PEN).

Parameter name: The name of the parameter.

Purpose: The purpose of the parameter. Give a short but clear
   description.

Description: Any special notes about the parameter, how it is to be
   used, etc.

Format definition: The ABNF for the parameter definition needs to be
   specified.

Example(s): One or more examples of instances of the parameter need
   to be specified.

10.2.5. Registration Template for Value Data Types

A value data type is defined by completing the following template.

Value name: The name of the value type.

Purpose: The purpose of the value type. Give a short but clear
   description.

Description: Any special notes about the value type, how it is to be
   used, etc.

Format definition: The ABNF for the value type definition needs to
   be specified.

Example(s): One or more examples of instances of the value type need
   to be specified.

10.2.6. Registration Template for Values

A value is defined by completing the following template.

Value: The value literal.

Purpose: The purpose of the value. Give a short but clear
   description.

Conformance: The vCard properties and/or parameters that can take
   this value needs to be specified.

Example(s): One or more examples of instances of the value need to
   be specified.

The following is a fictitious example of a registration of a vCard
value:

Value: supervisor

Purpose: It means that the related entity is the direct hierarchical
   superior (i.e., supervisor or manager) of the entity this vCard
   represents.

Conformance: This value can be used with the "TYPE" parameter
   applied on the "RELATED" property.

Example(s):

    RELATED;TYPE=supervisor:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

10.3. Initial vCard Elements Registries

The IANA has created and will maintain the following registries for
vCard elements with pointers to appropriate reference documents. The
registries are grouped together under the heading "vCard Elements".

10.3.1. Properties Registry

The following table has been used to initialize the properties
registry.

```
+-----------+-------------+------------------------+
| Namespace | Property    | Reference              |
+-----------+-------------+------------------------+
|           | SOURCE      | RFC 6350, Section 6.1.3 |
|           | KIND        | RFC 6350, Section 6.1.4 |
|           | XML         | RFC 6350, Section 6.1.5 |
|           | FN          | RFC 6350, Section 6.2.1 |
|           | N           | RFC 6350, Section 6.2.2 |
|           | NICKNAME    | RFC 6350, Section 6.2.3 |
|           | PHOTO       | RFC 6350, Section 6.2.4 |
|           | BDAY        | RFC 6350, Section 6.2.5 |
|           | ANNIVERSARY | RFC 6350, Section 6.2.6 |
|           | GENDER      | RFC 6350, Section 6.2.7 |
|           | ADR         | RFC 6350, Section 6.3.1 |
|           | TEL         | RFC 6350, Section 6.4.1 |
|           | EMAIL       | RFC 6350, Section 6.4.2 |
|           | IMPP        | RFC 6350, Section 6.4.3 |
|           | LANG        | RFC 6350, Section 6.4.4 |
|           | TZ          | RFC 6350, Section 6.5.1 |
|           | GEO         | RFC 6350, Section 6.5.2 |
|           | TITLE       | RFC 6350, Section 6.6.1 |
|           | ROLE        | RFC 6350, Section 6.6.2 |
|           | LOGO        | RFC 6350, Section 6.6.3 |
|           | ORG         | RFC 6350, Section 6.6.4 |
|           | MEMBER      | RFC 6350, Section 6.6.5 |
|           | RELATED     | RFC 6350, Section 6.6.6 |
|           | CATEGORIES  | RFC 6350, Section 6.7.1 |
|           | NOTE        | RFC 6350, Section 6.7.2 |
|           | PRODID      | RFC 6350, Section 6.7.3 |
|           | REV         | RFC 6350, Section 6.7.4 |
|           | SOUND       | RFC 6350, Section 6.7.5 |
|           | UID         | RFC 6350, Section 6.7.6 |
|           | CLIENTPIDMAP | RFC 6350, Section 6.7.7 |
|           | URL         | RFC 6350, Section 6.7.8 |
|           | VERSION     | RFC 6350, Section 6.7.9 |
|           | KEY         | RFC 6350, Section 6.8.1 |
|           | FBURL       | RFC 6350, Section 6.9.1 |
|           | CALADRURI   | RFC 6350, Section 6.9.2 |
|           | CALURI      | RFC 6350, Section 6.9.3 |
+-----------+-------------+------------------------+
```

10.3.2. Parameters Registry

The following table has been used to initialize the parameters
registry.

```
+-----------+-----------+-------------------------+
| Namespace | Parameter | Reference               |
+-----------+-----------+-------------------------+
|           | LANGUAGE  | RFC 6350, Section 5.1   |
|           | VALUE     | RFC 6350, Section 5.2   |
|           | PREF      | RFC 6350, Section 5.3   |
|           | ALTID     | RFC 6350, Section 5.4   |
|           | PID       | RFC 6350, Section 5.5   |
|           | TYPE      | RFC 6350, Section 5.6   |
|           | MEDIATYPE | RFC 6350, Section 5.7   |
|           | CALSCALE  | RFC 6350, Section 5.8   |
|           | SORT-AS   | RFC 6350, Section 5.9   |
|           | GEO       | RFC 6350, Section 5.10  |
|           | TZ        | RFC 6350, Section 5.11  |
+-----------+-----------+-------------------------+
```

10.3.3. Value Data Types Registry

The following table has been used to initialize the parameters
registry.

```
+-----------------+-------------------------+
| Value Data Type | Reference               |
+-----------------+-------------------------+
| BOOLEAN         | RFC 6350, Section 4.4   |
| DATE            | RFC 6350, Section 4.3.1 |
| DATE-AND-OR-TIME | RFC 6350, Section 4.3.4 |
| DATE-TIME       | RFC 6350, Section 4.3.3 |
| FLOAT           | RFC 6350, Section 4.6   |
| INTEGER         | RFC 6350, Section 4.5   |
| LANGUAGE-TAG    | RFC 6350, Section 4.8   |
| TEXT            | RFC 6350, Section 4.1   |
| TIME            | RFC 6350, Section 4.3.2 |
| TIMESTAMP       | RFC 6350, Section 4.3.5 |
| URI             | RFC 6350, Section 4.2   |
| UTC-OFFSET      | RFC 6350, Section 4.7   |
+-----------------+-------------------------+
```

10.3.4. Values Registries

Separate tables are used for property and parameter values.

The following table is to be used to initialize the property values
registry.

```
+----------+------------+-------------------------+
| Property | Value      | Reference               |
+----------+------------+-------------------------+
| BEGIN    | VCARD      | RFC 6350, Section 6.1.1 |
| END      | VCARD      | RFC 6350, Section 6.1.2 |
| KIND     | individual | RFC 6350, Section 6.1.4 |
| KIND     | group      | RFC 6350, Section 6.1.4 |
| KIND     | org        | RFC 6350, Section 6.1.4 |
| KIND     | location   | RFC 6350, Section 6.1.4 |
+----------+------------+-------------------------+
```

The following table has been used to initialize the parameter values
registry.

Property	Parameter	Value	Reference
FN, NICKNAME, PHOTO, ADR, TEL, EMAIL, IMPP, LANG, TZ, GEO, TITLE, ROLE, LOGO, ORG, RELATED, CATEGORIES, NOTE, SOUND, URL, KEY, FBURL, CALADRURI, and CALURI	TYPE	work	RFC 6350, Section 5.6
FN, NICKNAME, PHOTO, ADR, TEL, EMAIL, IMPP, LANG, TZ, GEO, TITLE, ROLE, LOGO, ORG, RELATED, CATEGORIES, NOTE, SOUND, URL, KEY, FBURL, CALADRURI, and CALURI	TYPE	home	RFC 6350, Section 5.6
TEL	TYPE	text	RFC 6350, Section 6.4.1
TEL	TYPE	voice	RFC 6350, Section 6.4.1
TEL	TYPE	fax	RFC 6350, Section 6.4.1
TEL	TYPE	cell	RFC 6350, Section 6.4.1
TEL	TYPE	video	RFC 6350, Section 6.4.1
TEL	TYPE	pager	RFC 6350, Section 6.4.1
TEL	TYPE	textphone	RFC 6350, Section 6.4.1
BDAY, ANNIVERSARY	CALSCALE	gregorian	RFC 6350, Section 5.8
RELATED	TYPE	contact	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	acquaintance	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	friend	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	met	RFC 6350, Section 6.6.6 and [xfn]

RELATED	TYPE	co-worker	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	colleague	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	co-resident	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	neighbor	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	child	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	parent	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	sibling	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	spouse	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	kin	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	muse	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	crush	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	date	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	sweetheart	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	me	RFC 6350, Section 6.6.6 and [xfn]
RELATED	TYPE	agent	RFC 6350, Section 6.6.6
RELATED	TYPE	emergency	RFC 6350, Section 6.6.6

11. Acknowledgments

The authors would like to thank Tim Howes, Mark Smith, and Frank
Dawson, the original authors of [RFC2425] and [RFC2426], Pete
Resnick, who got this effort started and provided help along the way,
as well as the following individuals who have participated in the
drafting, review, and discussion of this memo:

Aki Niemi, Andy Mabbett, Alexander Mayrhofer, Alexey Melnikov, Anil
Srivastava, Barry Leiba, Ben Fortuna, Bernard Desruisseaux, Bernie
Hoeneisen, Bjoern Hoehrmann, Caleb Richardson, Chris Bryant, Chris
Newman, Cyrus Daboo, Daisuke Miyakawa, Dan Brickley, Dan Mosedale,
Dany Cauchie, Darryl Champagne, Dave Thewlis, Filip Navara, Florian
Zeitz, Helge Hess, Jari Urpalainen, Javier Godoy, Jean-Luc Schellens,
Joe Hildebrand, Jose Luis Gayosso, Joseph Smarr, Julian Reschke,
Kepeng Li, Kevin Marks, Kevin Wu Won, Kurt Zeilenga, Lisa Dusseault,
Marc Blanchet, Mark Paterson, Markus Lorenz, Michael Haardt, Mike
Douglass, Nick Levinson, Peter K. Sheerin, Peter Mogensen, Peter
Saint-Andre, Renato Iannella, Rohit Khare, Sly Gryphon, Stephane
Bortzmeyer, Tantek Celik, and Zoltan Ordogh.

12. References

12.1. Normative References

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

[IEEE.754.2008]
          Institute of Electrical and Electronics Engineers,
          "Standard for Binary Floating-Point Arithmetic",
          IEEE Standard 754, August 2008.

[ISO.8601.2000]
          International Organization for Standardization, "Data
          elements and interchange formats - Information interchange
          - Representation of dates and times", ISO Standard 8601,
          December 2000.

[ISO.8601.2004]
          International Organization for Standardization, "Data
          elements and interchange formats - Information interchange
          - Representation of dates and times", ISO Standard 8601,
          December 2004.

   [RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail
              Extensions (MIME) Part One: Format of Internet Message
              Bodies", RFC 2045, November 1996.

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

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

   [RFC2739] Small, T., Hennessy, D., and F. Dawson, "Calendar
              Attributes for vCard and LDAP", RFC 2739, January 2000.

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

   [RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers",
              RFC 3966, December 2004.

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

   [RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally
              Unique IDentifier (UUID) URN Namespace", RFC 4122,
              July 2005.

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

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

   [RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322,
              October 2008.

   [RFC5545] Desruisseaux, B., "Internet Calendaring and Scheduling
              Core Object Specification (iCalendar)", RFC 5545,
              September 2009.

   [RFC5546] Daboo, C., "iCalendar Transport-Independent
              Interoperability Protocol (iTIP)", RFC 5546,
              December 2009.

   [RFC5646] Phillips, A. and M. Davis, "Tags for Identifying
              Languages", BCP 47, RFC 5646, September 2009.

   [RFC5870]   Mayrhofer, A. and C. Spanring, "A Uniform Resource
               Identifier for Geographic Locations ('geo' URI)",
                RFC 5870, June 2010.

   [RFC6351]   Perreault, S., "xCard: vCard XML Representation",
                RFC 6351, August 2011.

   [W3C.REC-xml-20081126]
               Maler, E., Yergeau, F., Sperberg-McQueen, C., Paoli, J.,
               and T. Bray, "Extensible Markup Language (XML) 1.0 (Fifth
               Edition)", World Wide Web Consortium Recommendation REC-
               xml-20081126, November 2008,
               <http://www.w3.org/TR/2008/REC-xml-20081126>.

   [xfn]   Celik, T., Mullenweg, M., and E. Meyer, "XFN 1.1 profile",
               <http://gmpg.org/xfn/11>.

12.2. Informative References

   [IANA-TZ]   Lear, E. and P. Eggert, "IANA Procedures for Maintaining
               the Timezone Database", Work in Progress, May 2011.

   [ISO9070]   International Organization for Standardization,
               "Information Processing - SGML support facilities -
               Registration Procedures for Public Text Owner
               Identifiers", ISO 9070, April 1991.

   [RFC1738]   Berners-Lee, T., Masinter, L., and M. McCahill, "Uniform
               Resource Locators (URL)", RFC 1738, December 1994.

   [RFC2397]   Masinter, L., "The "data" URL scheme", RFC 2397,
               August 1998.

   [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.

   [RFC3282]   Alvestrand, H., "Content Language Headers", RFC 3282,
               May 2002.

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

   [RFC3536]   Hoffman, P., "Terminology Used in Internationalization in
               the IETF", RFC 3536, May 2003.

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

   [RFC4880]   Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R.
               Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

   [RFC5335bis]
               Yang, A. and S. Steele, "Internationalized Email Headers",
               Work in Progress, July 2011.

   [RFC5751]   Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet
               Mail Extensions (S/MIME) Version 3.2 Message
               Specification", RFC 5751, January 2010.

   [RFC6068]   Duerst, M., Masinter, L., and J. Zawinski, "The 'mailto'
               URI Scheme", RFC 6068, October 2010.

   [TZ-DB]   Olson, A., "Time zone code and data",
               <ftp://elsie.nci.nih.gov/pub/>.

   [vCard21]   Internet Mail Consortium, "vCard - The Electronic Business
               Card Version 2.1", September 1996.

Appendix A. Differences from RFCs 2425 and 2426

This appendix contains a high-level overview of the major changes
that have been made in the vCard specification from RFCs 2425 and
2426. It is incomplete, as it only lists the most important changes.

A.1. New Structure

o [RFC2425] and [RFC2426] have been merged.

o vCard is now not only a MIME type but a stand-alone format.

o A proper MIME type registration form has been included.

o UTF-8 is now the only possible character set.

o New vCard elements can be registered from IANA.

A.2. Removed Features

o The CONTEXT and CHARSET parameters are no more.

o The NAME, MAILER, LABEL, and CLASS properties are no more.

o The "intl", "dom", "postal", and "parcel" TYPE parameter values
   for the ADR property have been removed.

o In-line vCards (such as the value of the AGENT property) are no
   longer supported.

A.3. New Properties and Parameters

o The KIND, GENDER, LANG, ANNIVERSARY, XML, and CLIENTPIDMAP
   properties have been added.

o [RFC2739], which defines the FBURL, CALADRURI, CAPURI, and CALURI
   properties, has been merged in.

o [RFC4770], which defines the IMPP property, has been merged in.

o The "work" and "home" TYPE parameter values are now applicable to
   many more properties.

o The "pref" value of the TYPE parameter is now a parameter of its
   own, with a positive integer value indicating the level of
   preference.

o The ALTID and PID parameters have been added.

   o The MEDIATYPE parameter has been added and replaces the TYPE
      parameter when it was used for indicating the media type of the
      property's content.

Author's Address

   Simon Perreault
   Viagenie
   2875 Laurier, suite D2-630
   Quebec, QC G1V 2M2
   Canada

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