Network Working Group
Request for Comments: 3625
Updates: 3555
Category: Informational

R. Gellens
H. Garudadri
Qualcomm

September 2003

The QCP File Format and Media Types for Speech Data

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

RFC 2658 specifies the streaming format for 3GPP2 13K vocoder (High Rate Speech Service Option 17 for Wideband Spread Spectrum Communications Systems, also known as QCELP 13K vocoder) data, but does not specify a storage format. Many implementations have been using the "QCP" file format (named for its file extension) for exchanging QCELP 13K data as well as Enhanced Variable Rate Coder (EVRC) and Selectable Mode Vocoders (SMV) data. (For example, Eudora(r), QuickTime(r), and cmda2000(r) handsets).

This document specifies the QCP file format and updates the audio/qcelp media registration to specify this format for storage, and registers the audio/evrc-qcp and audio/smv-qcp media types for EVRC and SMV (respectively) data stored in this format.

Table of Contents

1. Introduction

This document specifies the QCP format for storage of [QCELP] 13K, [EVRC], and [SMV] vocoder frames in files which may reside on network elements (e.g., media servers, Multimedia Messaging System (MMS) centers, etc.) and third-generation cellular (3G) handsets, or be transmitted via email or other mechanisms.

Vocoder frames read from a QCP file may be streamed using protocols and formats outside the scope of this document, delivered to an [EVRC], [SMV], or [QCELP] 13K decoder, or otherwise processed.

The QCP format supports [QCELP] 13K as well as [EVRC] and [SMV] data. Note that this format is different from the EVRC and SMV storage format specified in [EVRC-SMV].

2. Conventions Used in this Document

The key words "REQUIRED", "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as described in BCP 14, RFC 2119 [KEYWORDS].

3. QCP File Format

The following describes the QCP file format using [ABNF].

A qcp file contains data frames generated by the [QCELP], [EVRC], or [SMV] vocoders.

```
   qcp-file        = riff-qlcm fmt vrat [labl] [offs] data [cnfg]
                     [text]

   riff-qlcm       = RIFF riff-size QLCM

   fmt             = FMT chunk-size major minor codec-info
                     ; major and minor are set to the version of
                     ; the QCP file format used to generate the
                     ; file - currently, the following versions
                     ; are defined:
                     ;
                     ; major = 2, minor = 0
                     ;    used for SMV.
                     ; major = 1, minor = 0
                     ;    used for QCELP-13K and EVRC

   vrat            = VRAT chunk-size var-rate-flag size-in-packets

   labl            = LABL chunk-size label

   offs            = OFFS chunk-size step-size num-offsets *offset
                     ; number of repetitions of offset must equal
                     ; num-offsets

   data            = DATA chunk-size *packet [pad]
                     ; repetitions of packet contain the actual
                     ; packet data -- there should be as many
                     ; packets as indicated by the size-in-packets
                     ; element found elsewhere in this file format

   cnfg            = CNFG chunk-size config

   text            = TEXT chunk-size string [pad]
                     ; string contains any information defined
                     ; by the application

   riff-size       = UINT32
                     ; riff-size should equal total size of file
                     ; in bytes, excluding the size of RIFF and
                     ; riff-size

   chunk-size      = UINT32
                     ; chunk-size should equal the total size of
                     ; the chunk described by the rule containing
                     ; chunk-size, excluding the size of
                     ; chunk-size itself and any elements that
                     ; come before it in the rule, as well as
                     ; the pad element, if present
```

```
   major           = OCTET
                     ; major version number of QCP format --
                     ; currently set as "2" -- however, "1" should
                     ; be used instead for QCELP-13K and EVRC, to
                     ; maintain compatibility with older software
                     ; platforms

   minor           = OCTET
                     ; minor version number: currently set to "0"

   codec-info      = codec-guid codec-version codec-name
                            average-bps packet-size
                            block-size sampling-rate
                            sample-size variable-rate 5*UINT32
                     ; this identifies the codec used to encode
                     ; the speech in this file, and any parameters
                     ; needed in order to decode the speech

                     ; the five UINT32 elements are reserved for
                     ; use in future versions of this format,
                     ; should be set to zero for now, and readers
                     ; should allow non-zero values when reading
                     ; files in this format

   var-rate-flag   = UINT32
                     ; if zero, sizes of packets in data chunk are
                     ; fixed at the value indicated by the
                     ; packet-size field, found elsewhere in this
                     ; file format

                     ; if var-rate-flag is greater than zero but
                     ; less than %xFFFF0000, packets are variable
                     ; rate, and rate is determined by rate octet
                     ; in each packet -- to determine the size of
                     ; a packet, map the first octet (the rate
                     ; octet) to the size of the rest of the
                     ; packet, according to the rate-map-table
                     ; information found elsewhere in this file

                     ; values %xFFFF0001 and higher are reserved
                     ; for use in future versions of this format

   size-in-packets = UINT32
                     ; this is the total number of packets in the
                     ; data chunk in the file
```

```
   label           = 48OCTET
                     ; label is for generic storage for
                     ; application use

   step-size       = UINT32
                     ; difference in the times of sequential
                     ; application stored in this chunk, in
                     ; 100ms increments (step-size of 10 means
                     ; 1 second)

                     ; currently only a step-size of "10" is
                     ; guaranteed to be supported by software
                     ; capable of reading this file format

   num-offsets     = UINT32
                     ; number of offsets in offs chunk -- must
                     ; equal number of repetitions of offset
                     ; element

   offset          = UINT32
                     ; absolute octet offset in this QCP file
                     ; where the beginning of the packet that is
                     ; at a time index one step past the last
                     ; offset is located -- step size is given
                     ; in step-size

   packet          = [rate-octet] (1*OCTET)
                     ; the rate-octet element is present only if
                     ; the var-rate-flag found elsewhere in the file
                     ; indicates that the file contains
                     ; variable-rate packets - if it is present, the
                     ; octet is used to determine the size of the
                     ; remainder of the packet (the number of octets
                     ; which follow the rate-octet)

                     ; No rate-octet is present if the var-rate-flag
                     ; is set to zero, indicating a fixed size
                     ; packet file - in this case, the number of
                     ; octets in packet is equal to the value set
                     ; in packet-size, found elsewhere in this
                     ; file format

                     ; It should be noted that there is always a
                     ; codec rate octet in a packet, even if the
                     ; var-rate-flag is zero, indicating fixed size
                     ; packet files - in this case, although there
                     ; is no rate-octet element in this syntax, the
                     ; first octet in the packet is still considered
```

```
                    ; the "rate" for the packet.

                    ; Two examples follow to illustrate this.

                    ; Example 1.  Variable-rate QCELP
                    ; var-rate-flag = %d1
                    ; variable-rate = %d5 %d34 %d4 %d16 %d3 %d7 %d2
                    ;                 %d3 %d1 %d0 %d0 3(%d0 %d0)
                    ; packet        = %d4 (34OCTET)
                    ; packet        = %d4 (34OCTET)
                    ; packet        = %d3 (16OCTET)
                    ; packet        = %d1 (3OCTET)

                    ; Example 2.  Fixed-rate, full-rate QCELP
                    ; var-rate-flag = %d0
                    ; packet-size   = %d35
                    ; packet        = (35OCTET)  ; first octet is %d4
                    ; packet        = (35OCTET)  ; first octet is %d4
                    ; packet        = (35OCTET)  ; first octet is %d4
                    ; packet        = (35OCTET)  ; first octet is %d4

   pad              = %x00
                    ; pad is present only if the number of bytes in
                    ; the chunk described by the rule containing
                    ; pad would otherwise be odd - if present, its
                    ; size should NOT be included in the
                    ; calculation for any chunk-size element also
                    ; present in the chunk

   config           = UINT16
                    ; config is a bitmapped configuration word,
                    ; for application use

   string           = *(%x01-FF) %x00
                    ; this is a zero-terminated string of octets --
                    ; although not limited to it, typically the
                    ; string consists of us-ascii characters

   codec-guid       = UINT32 UINT16 UINT16 8OCTET
                    ; this is the unique identifier for the codec
                    ; used to encode the packets in the data chunk

                    ; The elements of this rule match the structure
                    ; defined for a GUID in other specifications
                    ; and formats.  The first three elements are
                    ; stored in little-endian octet order.
                    ; When values for a GUID are expressed, the
                    ; first three elements are expressed as a
```

```
                    ; sequence of hexadecimal digits in normal
                    ; network ordering (big-endian or most
                    ; significant digit first) while the eight
                    ; octet element is broken up into two groups,
                    ; the first having four hexadecimal digits,
                    ; and the second having twelve digits, all
                    ; expressed in network ordering.

                    ; For example, if the octets in a GUID as
                    ; stored in a file are:
                    ;   %12 %34 %56 %78 %9A %BC %DE %F0
                    ;   %0F %ED %CB %A9 %87 %65 %43 %21
                    ; then the GUID would have these values:
                    ;   UINT32 = %x78563412
                    ;   UINT16 = %xBC9A
                    ;   UINT16 = %xF0DE
                    ;   8OCTET = %x0F.ED.CB.A9.87.65.43.21
                    ; and the whole GUID would be expressed as:
                    ;   {78563412-BC9A-F0DE-0FED-CBA987654321}

                    ; The following codec GUIDs are currently
                    ; defined for QCP file format:
                    ;
                    ; QCELP-13K:
                    ;   {5E7F6D41-B115-11D0-BA91-00805FB4B97E}
                    ;   {5E7F6D42-B115-11D0-BA91-00805FB4B97E}
                    ; EVRC:
                    ;   {E689D48D-9076-46B5-91EF-736A5100CEB4}
                    ; SMV:
                    ;   {8D7C2B75-A797-ED49-985E-D53C8CC75F84}


   codec-version   = UINT16
                     ; version number of codec used to encode the
                     ; packets in the data chunk

                     ; This value depends on the particular codec
                     ; used to encode the packets.  The following
                     ; versions are currently defined:

                     ; QCELP-13K:
                     ;   1 or 2
                     ; EVRC, and SMV:
                     ;   1

   codec-name      = 80OCTET
                     ; the proper name of the codec, in us-ascii -
                     ; unused octets after the name are set to zero
```

```
   average-bps     = UINT16
                     ; average data rate, in bits per second, of
                     ; the speech data represented in this file

   packet-size     = UINT16
                     ; the size in octets of the largest possible
                     ; packet in the data chunk

   block-size      = UINT16
                     ; the number of samples encoded in every packet
                     ; in the data chunk

   sampling-rate   = UINT16
                     ; number of speech samples per second
                     ; (typically 8000)

   sample-size     = UINT16
                     ; number of bits per speech sample
                     ; (typically 16)

   variable-rate   = num-rates rate-map-table
                     ; if num-rates is zero, and major version
                     ; number of the QCP file is 2, then the
                     ; rate-map-table is not used - instead,
                     ; it is up to the decoder to determine the
                     ; sizes of packets in the data chunk, even
                     ; if var-rate-flag indicates the file uses
                     ; variable rate packets

                     ; otherwise, num-rates specifies how many
                     ; different possible rate octets there are
                     ; for the packets in the data chunk, and
                     ; thus how many of the rate-map-entry elements
                     ; contain valid information

   num-rates       = UINT32
                     ; this is the number of possible rate octets
                     ; used in the packets in the data chunk

   rate-map-table  = 8rate-map-entry
                     ; any unused entries SHOULD be filled with
                     ; %d0 %d0

   rate-map-entry  = rate-size rate-octet
                     ; this maps a possible rate octet for a packet
                     ; to the size of the rest of the packet having
                     ; that value for the rate octet
```

```
   rate-size       = OCTET
                     ; this is the size of a packet, excluding the
                     ; value for the rate octet

   rate-octet      = OCTET
                     ; this is the first octet of a packet in the
                     ; data chunk, when the var-rate-flag set for
                     ; the file indicates the file is variable rate

   UINT32          = 4OCTET
                     ; this field contains a 32-bit integer stored
                     ; as a sequence of four octets, in
                     ; little-endian order (least significant
                     ; octet first)

   UINT16          = 2OCTET
                     ; this field contains a 16-bit integer stored
                     ; as a sequence of two octets, in
                     ; little-endian order (least significant
                     ; octet first)

   OCTET           = %x00-FF
                     ; an octet, also called a byte - any possible
                     ; combination of eight bits, forming a single
                     ; integer or part of a larger integer having
                     ; more than eight bits

   RIFF            = %x52 %x49 %x46 %x46

   QLCM            = %x51 %x4C %x43 %x4D

   FMT             = %x66 %x6D %x74 %x20

   LABL            = %x6C %x61 %x62 %x6C

   OFFS            = %x6F %x66 %x66 %x73

   DATA            = %x64 %x61 %x74 %x61

   CNFG            = %x63 %x6E %x66 %x67

   TEXT            = %x74 %x65 %x78 %x74
```

4. IANA Considerations

IANA has updated the audio/qcelp and audio/vnd.qcelp registrations and has added the audio/evrc-qcp and audio/smv-qcp registrations as specified here.

4.1. Update of Media Type Audio/qcelp

The audio/qcelp media registration has been updated to indicate that this specification is to be used for storage.

4.2. Update of Media Type Audio/vnd.qcelp

The audio/vnd.qcelp media registration has been updated to indicate that use of this media type is deprecated and to note that the media type audio/qcelp should be used instead.

4.3. Registration of Audio/EVRC-QCP

Media Type Name: audio

Media Subtype Name: evrc-qcp

Required Parameter: none

Optional parameters: none

Encoding considerations:
The storage format specified in this document may be used with any transport mechanism.

Security considerations:
See Section 5 "Security Considerations" of this document.

Public specification: this document

Additional information: no

Magic number: First four octets: RIFF
Octets 9-12: QLCM

(Note: octets 5-8 constitute the riff-size field, which is the size of the file minus the RIFF header. Since this could be anything, it is not fixed and thus can not be used as part of the magic number.)

File extensions: qcp, QCP
Macintosh file type code: none
Object identifier or OID: none

Intended usage:
COMMON. This file format is already in wide use in Internet email user agents, multimedia authoring and playing software, and cdma2000(r) handsets.

Person & email address to contact for further information:
Harinath Garudadri hgarudad@qualcomm.com

Change controller:
The IETF

4.4. Registration of Audio/SMV-QCP

Media Type Name: audio

Media Subtype Name: smv-qcp

Required Parameter: none

Optional parameters: none

Encoding considerations:
The storage format specified in this document may be used with any transport mechanism.

Security considerations:
See Section 5 "Security Considerations" of this document.

Public specification: this document

Additional information: no

Magic number: First four octets: RIFF
Octets 9-12: QLCM

(Note: octets 5-8 constitute the riff-size field, which is the size of the file minus the RIFF header. Since this could be anything, it is not fixed and thus can not be used as part of the magic number.)

File extensions: qcp, QCP
Macintosh file type code: none
Object identifier or OID: none

Intended usage:
COMMON. This file format is already in wide use in Internet email user agents, multimedia authoring and playing software, and cdma2000(r) handsets.

Person & email address to contact for further information:
Harinath Garudadri hgarudad@qualcomm.com

Change controller:
The IETF

5. Security Considerations

This document specifies a file format only, not a streaming protocol payload format, nor a transfer method. As such, it introduces no security risks aside from those associated with any audio codec or media file format (for example, denial of service by transmitting a file larger than the receiver can handle). Note that those security concerns should be understood before using the file format specified here.

6. Acknowledgements

Richard Walters created the ABNF notation for this specification and proof-read the text, among other helpful tasks.

The qcp file format was originally developed by others within Qualcomm. The editor would like to thank Chuck Han and Livingstone Song for their contributions leading to this specification.

7. References

7.1. Normative References

[ABNF] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

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

[QCELP] 3GPP2 C.S0020 "High Rate Speech Service Option 17 for Wideband Spread Spectrum Communications Systems".

[EVRC] 3GPP2 C.S0014 "Enhanced Variable Rate Codec, Speech Service Option 3 for Wideband Spread Spectrum Digital Systems ". (Used to be TIA/EIA/IS-127)

[PureVoice] McKay, K., "RTP Payload Format for PureVoice(tm) Audio", RFC 2658, August 1999.

[SMV] 3GPP2 C.S0030 "Selectable Mode Vocoder, Service Option for Wideband Spread Spectrum Communication Systems".

7.2. Informative References

[EVRC-SMV] Li, A., "RTP Payload Format for Enhanced Variable Rate Codecs (EVRC) and Selectable Mode Vocoders (SMV)", RFC 3558, July 2003.

8. Intellectual Property Statement

9. Editors' Addresses

Randall Gellens
QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121
USA

EMail: randy@qualcomm.com

Harinath Garudadri
QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121
USA

EMail: hgarudad@qualcomm.com

10. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.