Unión Internacional de Telecomunicaciones

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

H.271

(05/2006)

SERIE H: SISTEMAS AUDIOVISUALES Y MULTIMEDIA

Infraestructura de los servicios audiovisuales – Codificación de imágenes vídeo en movimiento

Mensajes en el canal de retorno de vídeo para transmitir información de estado y solicitudes del receptor hacia el emisor de vídeo

Recomendación UIT-T H.271

Unión Internacional de Telecomunicaciones

RECOMENDACIONES UIT-T DE LA SERIE H

SISTEMAS AUDIOVISUALES Y MULTIMEDIA

CARACTERÍSTICAS DE LOS SISTEMAS VIDEOTELEFÓNICOS	H.100–H.199
INFRAESTRUCTURA DE LOS SERVICIOS AUDIOVISUALES	
Generalidades	H.200–H.219
Multiplexación y sincronización en transmisión	H.220–H.229
Aspectos de los sistemas	H.230–H.239
Procedimientos de comunicación	H.240–H.259
Codificación de imágenes vídeo en movimiento	**H.260–H.279**
Aspectos relacionados con los sistemas	H.280–H.299
Sistemas y equipos terminales para los servicios audiovisuales	H.300–H.349
Arquitectura de servicios de directorio para servicios audiovisuales y multimedia	H.350–H.359
Arquitectura de la calidad de servicio para servicios audiovisuales y multimedia	H.360–H.369
Servicios suplementarios para multimedia	H.450–H.499
PROCEDIMIENTOS DE MOVILIDAD Y DE COLABORACIÓN	
Visión de conjunto de la movilidad y de la colaboración, definiciones, protocolos y procedimientos	H.500–H.509
Movilidad para los sistemas y servicios multimedia de la serie H	H.510–H.519
Aplicaciones y servicios de colaboración en móviles multimedia	H.520–H.529
Seguridad para los sistemas y servicios móviles multimedia	H.530–H.539
Seguridad para las aplicaciones y los servicios de colaboración en móviles multimedia	H.540–H.549
Procedimientos de interfuncionamiento de la movilidad	H.550–H.559
Procedimientos de interfuncionamiento de colaboración en móviles multimedia	H.560–H.569
SERVICIOS DE BANDA ANCHA Y DE TRÍADA MULTIMEDIA	
Servicios multimedia de banda ancha sobre VDSL	H.610–H.619

Para más información, véase la Lista de Recomendaciones del UIT-T.

Recomendación UIT-T H.271

Mensajes en el canal de retorno de vídeo para transmitir información de estado y solicitudes del receptor hacia el emisor de vídeo

Resumen

La presente Recomendación especifica el formato de los mensajes en el canal de retorno para transmitir información de estado y solicitudes de un receptor de vídeo a un emisor de vídeo.

La sintaxis del mensaje se ha diseñado de manera genérica para que pueda utilizarse con la mayoría de las normas de codificación de vídeo internacionales existentes. Se especifica la aplicación de los mensajes genéricos conformes a las Recs. UIT-T H.261, H.263 y H.264 | ISO/CEI 14496-10.

Orígenes

La Recomendación UIT-T H.271 fue aprobada el 29 de mayo de 2006 por la Comisión de Estudio 16 (2005-2008) del UIT-T por el procedimiento de la Recomendación UIT-T A.8.

Palabras clave

Mensaje de canal de retorno, realimentación de receptor, selección de imagen de referencia, vídeo.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

La observancia de esta Recomendación es voluntaria. Ahora bien, la Recomendación puede contener ciertas disposiciones obligatorias (para asegurar, por ejemplo, la aplicabilidad o la interoperabilidad), por lo que la observancia se consigue con el cumplimiento exacto y puntual de todas las disposiciones obligatorias. La obligatoriedad de un elemento preceptivo o requisito se expresa mediante las frases "tener que, haber de, hay que + infinitivo" o el verbo principal en tiempo futuro simple de mandato, en modo afirmativo o negativo. El hecho de que se utilice esta formulación no entraña que la observancia se imponga a ninguna de las partes.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB en la dirección http://www.itu.int/ITU-T/ipr/.

© UIT 2006

Reservados todos los derechos. Ninguna parte de esta publicación puede reproducirse por ningún procedimiento sin previa autorización escrita por parte de la UIT.

ÍNDICE

			Página
1	Alcance		1
2	Referencias normativas		1
3	Definiciones		1
4	Abreviaturas, siglas o acrónimos		2
5	Convenios		2
	5.1	Operadores aritméticos	2
	5.2	Operadores lógicos	3
	5.3	Operadores relacionales	3
	5.4	Operadores de bit	3
	5.5	Operadores de asignación	4
	5.6	Variables, elementos sintácticos y cuadros	4
	5.7	Descripción mediante texto de operaciones lógicas	5
	5.8	Métodos de especificación de la sintaxis en forma tabular	6
	5.9	Especificación de la sintaxis de funciones, categorías y descriptores	7
6	Cabida útil de mensaje		8
	6.1	Sintaxis	8
	6.2	Semántica	10
7	Utilizaciones específicas normalizadas de los mensajes		12
	7.1	Utilizaciones específicas de los mensajes H.261	12
	7.2	Utilización específica de los mensajes H.263	12
	7.3	Utilización especifica de los mensajes H.264	14

Introducción

0.1 Objeto

Para ciertas aplicaciones, la transmisión de datos adicionales (dentro o fuera de la banda) es útil para mejorar la calidad de vídeo del servicio. Esta Recomendación especifica las cabidas útiles de datos para su utilización con una diversidad de tecnologías de codificación de vídeo. Las cabidas útiles de datos se especifican de manera genérica a fin de que sean aplicables a las normas de codificación de vídeo existentes más corrientes. Se especifica la aplicación de mensajes genéricos conformes a las Recs. UIT-T H.261, H.263 y H.264.

0.2 Consideraciones generales

Utilizando los mensajes por el canal de retorno definidos en la presente Recomendación, se puede señalar la siguiente información desde un receptor de vídeo hacia un emisor de vídeo:

– Informes de estado:
  - Una o más imágenes sin discordancia de error de tren de bit detectada.
  - Pérdidas de nivel de imagen y/o nivel de macrobloque.
  - Información de encabezamiento importante.

– Solicitudes de actualización:
  - Una solicitud de "reset" que indica que el emisor debe restaurar completamente el tren de bits de vídeo como si no hubiese recibido previamente ningún dato de tren de bits.

Se especifica la aplicación de mensajes en el canal de retorno conformes a las Recs. UIT-T H.261, H.263 y H.264.

Recomendación UIT-T H.271

Mensajes en el canal de retorno de vídeo para transmitir información de estado y solicitudes del receptor hacia el emisor de vídeo

1 Alcance

Esta Recomendación especifica los mensajes en el canal de retorno para codificación de vídeo basada en bloques.

2 Referencias normativas

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones, por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes. En esta Recomendación, la referencia a un documento, en tanto que autónomo, no le otorga el rango de una Recomendación.

– Recomendación UIT-T H.261 (1993), *Códec vídeo para servicios audiovisuales a p x 64 kbit/s*.

– Recomendación UIT-T H.263 (2005), *Codificación de vídeo para comunicación a baja velocidad binaria*.

– Recomendación UIT-T H.264 (2005), *Codificación de vídeo avanzada para los servicios audiovisuales genéricos*.

– ISO/CEI 14496-10:2005, *Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding*.

3 Definiciones

En esta Recomendación se definen los términos siguientes.

3.1 canal de retorno: Medio para transportar mensajes por canal de retorno desde un receptor de tren de bits de vídeo hacia un emisor de tren de bits de vídeo.

3.2 mensaje de canal de retorno: Mensaje generado por un receptor de un *tren de bits* de vídeo que transporta solicitudes o información de estado del receptor.

3.3 tren de bits: Secuencia de bits que forma la representación de *imágenes* y los datos asociados codificados y forman una o más secuencias de vídeo codificadas.

3.4 error de tren de bits: *Tren de bits* contaminado incompleto.

3.5 desajuste de error de tren de bits: Diferencia entre los valores o la cantidad de *imágenes* decodificadas causada por una o más *errores de tren de bits* y los valores o la cantidad de *imágenes* descodificadas generadas por el proceso de descodificación con un *tren de bits* sin *errores de tren de bits*.

3.6 bloque: Matriz MxN (M-columnas por N-filas) de muestras de luminancia y las muestras de crominancia asociadas.

3.7 error de tren de bits detectado: *Error de tren de bits* que detecta el receptor.

3.8 desajuste por errores de tren de bits detectados: *Desajuste por error de tren de bits* que puede producirse debido a *errores de tren de bits*.

3.9 orden de decodificación: Orden en el que el proceso de decodificación especificado por una tecnología de codificación de vídeo procesa los *elementos sintácticos.*

3.10 macrobloque: *Bloque* de 16x16 muestras de luminancia y sus dos bloques correspondientes de muestras de crominancia.

3.11 conjunto de parámetros: Estructura sintáctica que contiene un número de *elementos sintácticos* que se pueden utilizar en el proceso de decodificación para una o más *imágenes*.

3.12 imagen: Término genérico que designa un campo o un marco de vídeo codificado como una unidad separada por una tecnología de codificación de vídeo.

3.13 imagen de referencia: *Imagen* que contiene muestras que se pueden utilizar para la predicción ínter en el proceso de decodificación de las subsiguientes *imágenes* en el tren de bits de vídeo.

3.14 reservado: Cuando se utiliza este término en las cláusulas que especifican los valores de un elemento sintáctico determinado, significa que el valor está reservado para un uso futuro en el UIT-T. Si bien esos valores no se utilizan en los mensajes por canal de retorno conformes con la presente Recomendación, es posible que el UIT-T los utilice en futuras Recomendaciones.

3.15 elemento sintáctico: Elemento de datos representado en el *tren de bits* o en un mensaje por canal de retorno.

4 Abreviaturas, siglas o acrónimos

En esta Recomendación se utilizan las siguientes abreviaturas, siglas o acrónimos.

CRC Código por redundancia cíclica (*cyclic redundancy code*)

LSB Bit menos significativo (*least significant bit*)

MSB Bit más significativo (*most significant bit*)

5 Convenios

En la presente Recomendación, los párrafos que aparecen con el preámbulo "NOTA –" tienen carácter informativo y no forman parte integrante de esta Recomendación.

NOTA – Los operadores matemáticos utilizados en esta Recomendación son similares a los del lenguaje de programación C. Convencionalmente, al enumerar y contar se empieza generalmente desde 0.

5.1 Operadores aritméticos

A continuación, se definen los siguientes operadores aritméticos.

+ Suma

– Resta (cuando se especifican dos argumentos) o negación (como operador de prefijo unario)

* Multiplicación

/ División entera en la que se descartan los decimales. Por ejemplo, 7/4 y (–7) / (–4) se trunca al valor 1 y (–7) / 4 y 7 / (–4) se truncan al valor –1.

x % y Módulo. Resto de dividir x por y, definido sólo para valores enteros de x e y, siendo $x \geq 0$ e $y > 0$.

Si no se utilizan paréntesis para indicar explícitamente el orden de precedencia, se aplican las siguientes reglas:

– la multiplicación y la división se realizan antes que la suma y la resta;

– la multiplicación y la división en una expresión se calculan por orden de izquierda a derecha;

– la suma y la resta en una expresión se calculan por orden de izquierda a derecha.

5.2 Operadores lógicos

Se definen los siguientes operadores lógicos:

x && y función lógica booleana "y" entre x e y

x | | y función lógica booleana "o" entre x e y

! función lógica booleana "no"

x ? y : z si x es TRUE (verdadero) o distinto de 0, toma el valor de y; en caso contrario, toma el valor de z.

5.3 Operadores relacionales

Se definen los siguientes operadores relacionales:

> mayor que

>= mayor o igual que

< menor que

<= menor o igual que

= = igual a

!= distinto de

5.4 Operadores de bit

Se definen los siguientes operadores de bit:

& Función lógica "y" bit a bit. Cuando se aplica a números enteros, se toma la representación en complemento a dos de los números enteros. Cuando se aplica entre dos números binarios con distinto número de bits, al más corto se le añaden ceros a la izquierda, se rellenan con 0 los bits más significativos.

| Función lógica "o" bit a bit. Cuando se aplica a números enteros, se toma la representación en complemento a dos de los números enteros. Cuando se aplica entre dos números binarios con distinto número de bits, al más corto se le añaden ceros a la izquierda, se rellenan con 0 los bits más significativos.

^ Función lógica "o exclusiva" bit a bit. Cuando se aplica a números enteros, se toma la representación en complemento a dos de los números enteros. Cuando se aplica entre dos números binarios con distinto número de bits, al más corto se le añaden ceros a la izquierda, se rellenan con 0 los bits más significativos.

x >> y Desplaza y posiciones hacia la derecha los bits de x, estando x expresado en complemento a dos. Esta función sólo esta definida para valores enteros positivos de y. El valor de los MSB de x es el mismo antes y después del desplazamiento.

x << y Desplaza y posiciones hacia la izquierda los bits de x, estando x expresado en complemento a dos. Esta función sólo esta definida para valores enteros positivos de y. Al aplicar el desplazamiento, los bits añadidos en los LSB tiene un valor igual a 0.

5.5 Operadores de asignación

Se definen los siguientes operadores de asignación:

= Operador de asignación.

+ + Aumento es decir, x++ es equivalente a x = x + 1; cuando se utiliza en un índice de una matriz, el valor de la variable se toma antes de aplicar el operador aumento.

– – Disminución es decir, x– – es equivalente a x = x – 1; cuando se utiliza en un índice de una matriz, el valor de la variable se toma antes de aplicar el operador disminución.

+= Aumento por un valor determinado es decir, x += 3 es equivalente a x = x + 3, y x += (–3) es equivalente a x = x + (–3).

– = Disminución por un valor determinado es decir, x – = 3 es equivalente a x = x – 3, y x – = (–3) es equivalente a x = x – (–3).

5.6 Variables, elementos sintácticos y cuadros

Los elementos sintácticos en el tren de bits se representan en **negritas**. Cada elemento sintáctico se describe por su nombre (en minúsculas y con el carácter subrayado), su categoría o sus dos categorías sintácticas, y uno o dos descriptores de su método de representación codificada. El proceso de decodificación se lleva a cabo de acuerdo con el valor del elemento sintáctico y de los valores de los elementos sintácticos decodificados previamente. Cuando el valor de un elemento sintáctico se utiliza en los cuadros de sintaxis o en el texto, aparece en letra normal (es decir, no está en negritas).

Quizá en algunos casos los cuadros sintácticos utilicen los valores de otras variables derivadas de los valores de los elementos sintácticos. Estas variables aparecen en los cuadros de sintaxis o en el texto en una combinación de mayúsculas y minúsculas y sin el carácter subrayado. Las variables que empiecen con mayúscula se obtienen para la decodificación de la estructura sintáctica considerada y todas las estructuras sintácticas dependientes. Las variables que empiezan con mayúscula se pueden utilizar en el proceso de decodificación de estructuras sintácticas posteriores que mencionen la estructura sintáctica original de la variable. Las variables que empiezan con minúscula sólo se utilizan en la subcláusula en que tiene su origen.

En algunos casos, se utilizan indistintamente nombres "mnemónicos" o sus valores numéricos para referirse a los valores de los elementos sintácticos o valores variables. Algunas veces se utilizan nombres "mnemónicos" que no tienen asociados valores numéricos. La relación entre valores y nombres se especifican en el texto. Los nombres están formados por un grupo de letras o varios grupos separados por un carácter subrayado. Cada grupo empieza con mayúscula aunque puede contener varias letras mayúsculas.

NOTA – La sintaxis se describe de manera muy similar a la estructura sintáctica del lenguaje C.

Las funciones se describen por su nombre, cuya forma es la de los nombres de elementos sintácticos, y entre paréntesis encierra cero o los nombres de las variables (para su definición) o valores (para su utilización), separados por comas (en caso de que haya más de una variable).

La notación binaria se indica poniendo la cadena de bits entre comillas sencillas. Por ejemplo, '01000001' representa una cadena de ocho bits en la que sólo vale 1 el segundo bit y el último.

Cuando el número de bits es un múltiplo entero de cuatro, en lugar de la notación binaria se puede utilizar notación hexadecimal, que se indica mediante el prefijo "0x", antes del número hexadecimal. Por ejemplo, 0x41 representa una cadena de ocho bits en la que sólo vale 1 el segundo bit y el último.

Los valores numéricos que no están entre comillas sencillas y que no tienen el prefijo "0x" son valores decimales.

El valor 0 representa la condición FALSE (FALSO) en una expresión condicional. El valor TRUE (VERDADERO) se representa por cualquier otro valor distinto de cero.

5.7 Descripción mediante texto de operaciones lógicas

La expresión de operaciones lógicas que en seudocódigo se describirían como:

```
if( condición 0 )
  expresión 0
else if ( condición 1 )
 expresión 1
…
else /* observación sobre las condiciones restantes */
  expresión n
```

en el texto se escribe del modo siguiente:

... del modo siguiente /... se aplica lo siguiente.

– Si condición 0, expresión 0
– De lo contrario, si condición 1, expresión 1
– …
– De lo contrario (observación sobre las condiciones restantes), expresión n

Cada expresión "Si...de lo contrario, si...de lo contrario, ..." en el texto empieza con "... del modo siguiente" o "... se aplica lo siguiente" seguido inmediatamente de "Si ...". La última condición del "Si...de lo contrario, si... de lo contrario, ..." siempre es un "De lo contrario, ...". El intercalado de expresiones "Si...de lo contrario, si...de lo contrario, ..." se puede identificar por que empieza con "... del modo siguiente " o "... se aplica lo siguiente" y termina con el último "De lo contrario, ...".

La expresión de operaciones lógicas que en seudocódigo se describirían mediante:

```
if( condición 0a && condición 0b )
  expresión 0
else if (condición 1a | | condición 1b )
  expresión 1
…
else
  expresión n
```

en el texto se escribe del siguiente modo:

... del modo siguiente / ... se aplica lo siguiente.

– Si se cumplen todas las condiciones siguientes, expresión 0
  - condición 0a
  - condición 0b
– De lo contrario, si se cumple alguna de las siguientes condiciones, expresión 1
  - condición 1a
  - condición 1b
– …
– De lo contrario, expresión n

La expresión de operaciones lógicas que en seudocódigo se escriben como

```
if( condición 0 )
  expresión 0
if( condición 1 )
  expresión 1
```

en el texto se escriben de la siguiente manera:

si condición 0, expresión 0

si condición 1, expresión 1

5.8 Métodos de especificación de la sintaxis en forma tabular

El siguiente cuadro muestra ejemplos de seudocódigo utilizado para describir la sintaxis. Cuando aparece **syntax_element** (elemento sintáctico), éste especifica que el elemento de sintaxis se analiza del tren de bits y el puntero del tren de bits avanza a la posición siguiente después del elemento sintáctico en el proceso de análisis del tren de bits.

	Descriptor
/* Una expresión puede ser un elemento sintáctico con una categoría sintáctica asociada y un descriptor o puede ser una expresión que especifique condiciones de existencia, de tipo y cuantitativas de elementos sintácticos, como los dos ejemplos siguientes. */	
syntax_element	ue(v)
expresión condicional o estructura {	
/* Un grupo de expresiones encerradas entre llaves es una expresión compuesta y funcionalmente se trata como si fuera una expresión sencilla */	
expresión	
expresión	
…	
}	
/* La estructura "while" es una expresión condicional que especifica la comprobación de una condición, y si ésta es verdadera especifica la expresión (o la expresión compuesta) que se repite hasta que la condición deja de ser verdadera */	
while(condición)	
expresión	
/* La estructura "do … while" es una estructura condicional que especifica una expresión que se aplica una vez, seguido de una comprobación de una condición, y si ésta es verdadera se repite la expresión anterior hasta que la condición deja de ser verdadera */	
do	
expresión	
while(condición)	

	Descriptor
/* La estructura "if … else" especifica la comprobación de una condición, y si ésta es verdadera especifica una expresión primaria, de lo contrario especifica una expresión alternativa. La parte "else" de la estructura y la expresión alternativa se omiten en caso de que ésta no sea necesaria */	
if(condición)	
expresión primaria	
else	
expresión alternativa	
/* La estructura "for" especifica una expresión inicial, seguida de la comprobación de una condición que, si es verdadera, especifica una expresión primaria que se repite seguida de una expresión subsiguiente hasta que la condición deja de ser verdadera */	
for(expresión inicial; condición; expresión subsiguiente)	
expresión primaria	

5.9 Especificación de la sintaxis de funciones, categorías y descriptores

Aquí se describen las funciones que se utilizan en las descripciones sintácticas. En las funciones se supone que existe un puntero al tren de bits que indica la posición del siguiente bit que leerá del tren de bits el proceso de codificación.

byte_aligned( ) se define del modo siguiente:

– Si la posición actual en el tren de bits es un límite de byte, es decir, el siguiente bit en el tren de bits es el primer bit de un byte, el valor que devuelve byte_aligned( ) es TRUE (VERDADERO).

– De lo contrario, el valor que devuelve byte_aligned( ) es FALSE (FALSO).

more_msg_data( ) se define del modo siguiente:

– Si en la estructura sintáctica msg_data( ), hay más datos, el valor que devuelve more_msg_data( ) es TRUE (VERDADERO).

– De lo contrario, el valor que devuelve more_msg_data( ) es FALSE (FALSO).

El método para rehabilitar la determinación de si hay más datos en la estructura sintáctica msg_data( ) depende de la aplicación.

next_bits(n) lee los siguientes bits en el flujo de bits con objeto de hacer comparaciones, sin avanzar el puntero al tren de bits. Se utiliza para leer los siguientes n bits del tren de bits, siendo n su argumento.

read_bits(n) lee los siguientes n bits del tren de bits y avanza el puntero al tren de bits n posiciones de bit. Cuando n es igual a 0, read_bits(n) devuelve 0 y no avanza el puntero al tren de bits.

A continuación se describen los descriptores para el proceso de análisis sintáctico de cada elemento sintáctico.

– f(n): cadena de bits de patrón fijo que utiliza n bits escritos (de izquierda a derecha), siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor queda especificado por el valor que devuelve la función read_bits(n).

– u(n): entero sin signo de n bits. Cuando n es "v" en el cuadro de sintaxis, el número de bits varía en función del valor de otros elementos sintácticos. El proceso de análisis sintáctico para este descriptor queda especificado por el valor que devuelve la función read_bits(n), valor que se interpreta como la representación binaria de un entero sin signo, siendo el bit más significativo el primero.

– ue(v): elemento sintáctico codificado mediante codificación Exp-Golomb entero sin signo, siendo el primer bit el izquierdo. El proceso de análisis sintáctico para este descriptor se describe a continuación.

Los elementos sintácticos codificados como ue(v) están codificados con Exp-Golomb. El proceso de análisis sintáctico de esos elementos sintácticos empieza con la lectura de bits a partir de la posición actual en el tren de bits hasta el primer bit distinto de 0, inclusive, y se cuenta el número de bits que están primero y son iguales a 0. Este proceso se especifica como sigue:

```
leadingZeroBits = –1;
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )
```

La variable codeNum se asigna mediante la siguiente expresión:

$$codeNum = 2^{leadingZeroBits} - 1 + read_bits(leadingZeroBits)$$

donde el valor que devuelve read_bits(leadingZeroBits) se interpreta como una representación binaria de un entero sin signo de modo que el primer bit escrito es el más significativo.

El valor del elemento sintáctico es codeNum.

6 Cabida útil de mensaje

6.1 Sintaxis

msg_data( ) {	**Descriptor**
do	
message( )	
while(more_msg_data( ))	
}	

message( ) {	**Descriptor**
payloadType = 0	
while(next_bits(8) = = 0xFF) {	
ff_byte /* equal to 0xFF */	f(8)
payloadType += 255	
}	
last_payload_type_byte	u(8)
payloadType += last_payload_type_byte	
payloadSize = 0	
while(next_bits(8) = = 0xFF) {	
ff_byte /* equal to 0xFF */	f(8)
payloadSize += 255	
}	
last_payload_size_byte	u(8)
payloadSize += last_payload_size_byte	
msg_payload(payloadType, payloadSize)	
}	

msg_payload(payloadType, payloadSize) {	**Descriptor**
if(payloadType <= 4)	
ref_pic_id	**u(32)**
if(payloadType = = 0) {	
num_ref_pics_minus1	**ue(v)**
for(i = 1; i <= num_ref_pics_minus1; i++)	
good_ref_pic_id[i]	u(32)
} else if(payloadType = = 1)	
delta_ref_pic_id	**ue(v)**
else if(payloadType = = 2) {	
data_partition_idc	ue(v)
run_length_flag	u(1)
if(run_length_flag) {	
first_blk_lost	ue(v)
num_blks_lost_minus1	ue(v)
} else {	
top_left_blk	ue(v)
bottom_right_blk	ue(v)
}	
} else if((payloadType = = 3) || (payloadType = = 4)) {	
param_set_type	ue(v)
param_set_crc	u(16)

if(payloadType = = 3)	
param_set_id	ue(v)
}	
stop_one_bit /* equal to 1 */	f(1)
while(!byte_aligned( ))	
alignment_zero_bit /* equal to 0 */	f(1)
}	

6.2 Semántica

ff_byte será igual a 0xFF.

last_payload_type_byte es el último byte utilizado para especificar el tipo de cabida útil de un mensaje.

last_payload_size_byte es el último byte utilizado para especificar las dimensiones de un mensaje.

Las dimensiones de la estructura sintáctica msg_payload( ) serán iguales a payloadSize * 8 bits.

Los tipos de mensaje que corresponden a los valores de payloadType figuran en el cuadro 6-1.

Cuadro 6-1/H.271 – Tipos de mensaje

payloadType	Mensaje
0	Una o más imágenes sin desajuste de error de tren de bits detectado.
1	Una o más imágenes total o parcialmente perdidas.
2	Un conjunto de bloques de una imagen total o parcialmente perdida.
3	CRC para un conjunto de parámetros.
4	CRC para todos los conjuntos de parámetros de un cierto tipo.
5	Una solicitud de "reiniciación" que indica que el emisor debe restaurar completamente el tren de bits de vídeo como si no se hubiese recibido previamente ningún dato de tren de bits.
> 5	Reservado para un uso futuro por el UIT-T.

Los mensajes cuyo payloadType es superior a 5 se suprimen y descartan según las dimensiones del mensaje conforme a lo que se indica en payloadSize.

ref_pic_id identifica una imagen. En función del valor de payloadType, la semántica de ref_pic_id se especifica en el cuadro 6-2.

Cuadro 6-2/H.271 – Semántica de ref_pic_id

payloadType	Semántica de ref_pic_id
0	Especifica una imagen sin desajuste de error de tren de bit detectado.
1	Especifica una imagen que se ha totalmente o parcialmente perdido.
2	Especifica una imagen que se ha parcialmente perdido.
3	Especifica una imagen. Para un conjunto de parámetros asociados con la imagen especificada, hay un CRC.
4	Especifica una imagen. Para todos los conjuntos de parámetros de un cierto tipo que se almacenan en un momento asociado con la decodificación de la imagen especificada, hay un CRC.

num_ref_pics_minus1 más 1 especifica el número de imágenes sin desajuste de error de tren de bits detectado. El valor de num_ref_pics_minus1 estará comprendido entre 0 y 31 inclusive.

good_ref_pic_id[i] especifica la i[a] la imagen indicada en el mensaje actual sin desajuste de error de tren de bits detectado.

delta_ref_pic_id especifica un aumento relativo a ref_pic_id que identifica un conjunto de imágenes que se han totalmente o parcialmente perdido. El valor de delta_ref_pic_id estará comprendido entre 0 y 31 inclusive.

data_partition_idc indica que todos los datos (cuando data_partition_idc vale 0) o una partición de los datos (cuando data_partition_idc es distinto de 0) para los bloques especificados por top_left_blk y bottom_right_blk se han perdido. El valor de data_partition_idc estará comprendido entre 0 y 15 inclusive.

run_length_flag igual a 1 indica la presencia de los elementos sintácticos first_blk_lost y num_blks_lost_minus1. run_length_flag igual a 0 indica la presencia de los elementos sintácticos top_left_blk y bottom_right_blk.

first_blk_lost indica la dirección de bloque del primer bloque en el orden de barrido. La dirección de bloque es el índice de un bloque en un barrido de bloque de la imagen que comienza con cero para el bloque superior izquierdo en una imagen.

num_blks_lost_minus1 más 1 especifica el número de bloques consecutivos en orden de barrido con el primer bloque identificado por first_blk_lost.

top_left_blk y **bottom_right_blk** especifican las direcciones de los bloques superior izquierdo e inferior derecho, respectivamente, de la región rectangular que se ha perdido total o parcialmente. Cuando la variable PicWidthInBlks es la anchura de imagen en unidades de bloques, los valores de los elementos sintácticos top_left_blk y bottom_right_blk deberán seguir las siguientes pautas.

– top_left_blk será menor o igual que bottom_right_blk.

– bottom_right_blk será menor que el tamaño de la imagen en unidades de bloques.

– top_left_blk % PicWidthInBlks será menor o igual que bottom_right_blk % PicWidthInBlks.

param_set_type especifica el tipo de conjuntos de parámetros. El valor de param_set_type estará comprendido entre 0 y 15 inclusive.

param_set_crc es el CRC para el conjunto de parámetros identificado por param_set_id y param_set_type cuando payloadType es igual a 3 y es el CRC para todos los conjuntos de parámetros del tipo especificado por param_set_type juntos cuando payloadType es igual a 4.

El valor de param_set_crc será igual al valor de crcVal que se obtiene efectuando el siguiente proceso de seudocódigo.

```
paramSet[pLen     ] = 0
paramSet[pLen + 1] = 0
crcVal = 0xFFFF
for( bitIdx = 0; bitIdx < ( pLen + 2 ) * 8; bitIdx++ ) {
   crcMsb = ( crcVal >> 15 ) & 1
   bitVal = ( paramSet[bitIdx >> 3] >> ( 7 – ( bitIdx & 7 ) ) ) & 1
   crcVal = ( ( ( crcVal << 1 ) + bitVal ) & 0xFFFF ) ^ ( crcMsb * 0x1021 )
}                                                                    (6-1)
```

donde la variable paramSet es una cadena de bytes que contiene al comienzo los datos para los cuales se calcula el CRC; la variable pLen es el número de bytes de los datos para los cuales se calcula el CRC; y la cadena de bytes de la variable paramSet tiene una longitud suficiente como

para añadir al final de los datos para los cuales se calcula el CRC dos bytes adicionales de valor cero.

Cuando payloadType es igual a 3, paramSet contiene, en el orden de los bytes de red, los bytes del conjunto de parámetros recibido identificado por param_set_id y param_set_type.

Cuando payloadType es igual a 4, paramSet contiene una concatenación, en orden de bytes de red, los bytes de todos los conjuntos de parámetros del tipo identificado por param_set_type, por orden creciente de identificador de conjunto de parámetros. Para todos los conjuntos de parámetros que nunca han sido recibidos, se considerará que la representación del conjunto de parámetros tiene una longitud de dos bytes, con el primer byte que contiene los ocho MSB y el segundo byte que contiene los ocho LSB de una representación binaria de 16 bits sin signo del valor del identificador de conjunto de parámetros.

param_set_id especifica el identificador de conjunto de parámetros del conjunto de parámetros del tipo especificado por param_set_type para los cuales hay un CRC. El valor de param_set_id estará comprendido entre 0 y 65535 inclusive.

stop_one_bit será igual a 1.

alignment_zero_bit será igual a 0.

7 Utilizaciones específicas normalizadas de los mensajes

En esta cláusula, bit n de una variable se refiere al n-ésimo bit menos significativo de la representación binaria de la variable. El LSB se computa como bit 0.

7.1 Utilizaciones específicas de los mensajes H.261

El valor de payloadType será igual a 0, 1, 2 ó 5. Los mensajes con otros valores de payloadType serán ignorados.

Cuando haya ref_pic_id o good_ref_pic_id[i], los 5 LSB de esos elementos sintácticos indicarán el valor de TR, que será igual a ref_pic_id & 0x1F o good_ref_pic_id[i] & 0x1F, de una imagen. Los bits restantes de ref_pic_id o good_ref_pic_id[i], que se reservan para una futura utilización por el UIT-T, serán iguales a 0 y serán ignorados.

Según el valor de payloadType, se aplicará lo siguiente.

– Si payloadType es igual a 0, la imagen que tiene TR igual a ref_pic_id & 0x1F o good_ref_pic_id[i] & 0x1F no tiene desajuste por error de tren de bit detectado.

– De lo contrario, si payloadType es igual a 1, todas las imágenes, en orden de decodificación, comenzando con una imagen que tiene TR igual a ref_pic_id & 0x1F, hasta una imagen (incluida) que tiene TR igual a ((ref_pic_id & 0x1F) + delta_ref_pic_id) % 32, cuando esas imágenes están presentes en el tren de bits de vídeo, se indican como total o parcialmente perdidas.

– Por otra parte, si payloadType es igual a 2, la imagen cuyo TR es igual a ref_pic_id & 0x1F ha sido parcialmente perdida. Cada bloque es un macrobloque. El valor de data_partition_idc será igual a 0, lo que significa que todos los datos de los macrobloques identificados se consideran perdidos. Todos los demás valores de data_partition_idc se reservan para una futura utilización por el UIT-T y serán ignorados.

– Si (payloadType es igual a 5), no hay más especificaciones.

7.2 Utilización específica de los mensajes H.263

Entre las características de H.263 soportadas por la presente sintaxis figuran las siguientes:

– Imágenes de larga duración.

– Partición de datos.

Las siguientes características de H.263 no están soportadas por la presente sintaxis:

– Campos entrelazados.

El valor de payloadType será igual a 0, 1, 2 ó 5. Se ignorarán los mensajes con otros valores de payloadType.

Cuando hay ref_pic_id o good_ref_pic_id[i], los 12 LSB indican el valor de la variable picIdentifier, que es igual a ref_pic_id & 0x0FFF o good_ref_pic_id[i] & 0x0FFF. Cuando se utiliza el anexo U/H.263 y payloadType es igual a 0, el bit 12 indica si la imagen almacenada es una imagen de largo plazo (cuando el bit es igual 1) o una imagen de corto plazo (cuando el bit es igual 0). El bit 13 será igual a 0 a menos que se utilice el modo de escalabilidad temporal, SNR y espacial opcional (anexo O/H.263). Cuando el bit 13 es igual a 1, el mensaje se refiere a una capa de mejora (más que a la capa básica), y los bits 14 a 17 inclusive, contienen el valor de ELNUM, que es igual a ref_pic_id & 0x03C000 o good_ref_pic_id[i] & 0x03C000. Los bits restantes de ref_pic_id o good_ref_pic_id[i], que se reservan para una futura utilización por el UIT-T, serán iguales a 0 y serán ignorados.

– Si no se utiliza el anexo U/H.263, picIdentifier indica el valor de TR de una imagen. Ese valor de TR será menor que MaxTR que es igual al valor máximo posible de TR más 1. Según el valor de payloadType, se aplica lo siguiente:

  • Si payloadType es igual a 0, el bit 12 de ref_pic_id o good_ref_pic_id[i] será igual a 0. La imagen cuyo TR es igual a picIdentifier, y cuando el bit 13 de ref_pic_id o good_ref_pic_id[i] es igual a 1, con ELNUM igual a ref_pic_id & 0x03C000 o good_ref_pic_id[i] & 0x03C000, no tiene desajuste por error de tren de bits detectado.

  • De lo contrario, si payloadType es igual 1, el bit 12 de ref_pic_id será igual a 0. Todas las imágenes, en orden de decodificación y en la capa básica o la misma capa de mejora con ELNUM igual a ref_pic_id & 0x03C000, que comienzan con una imagen cuyo TR es igual a ref_pic_id & 0x0FFF, hasta una imagen (incluida) con TR igual a ((ref_pic_id & 0x0FFF) + delta_ref_pic_id) % MaxTR, cuando esas imágenes se encuentran en el tren de bits de vídeo, se indica que han sido total o parcialmente perdidas.

  • Por otra parte, si payloadType es igual a 2, el bit 12 de ref_pic_id será igual a 0. La imagen con TR igual a picIdentifier y, cuando el bit 13 de ref_pic_id es igual a 1, con ELNUM igual a ref_pic_id & 0x03C000, ha sido parcialmente perdida. El valor de data_partition_idc indica que todos los datos (cuando data_partition_idc es igual a 0), partición de datos de encabezamiento (cuando data_partition_idc es igual a 1), partición del vector de movimiento (cuando data_partition_idc es igual a 2) o partición de datos de coeficiente (cuando data_partition_idc es igual a 3) de los macrobloques identificados se han perdido. Todos los demás valores de data_partition_idc se reservan para uso futuro por el UIT-T y serán ignorados.

  • Si (payloadType es igual a 5), no hay más especificaciones.

– De lo contrario (se utiliza el anexo U/H.263), según el valor del tipo de cabida útil, se aplica lo siguiente:

  • Si payloadType es igual a 0, ref_pic_id o good_ref_pic_id[i] identifica una imagen almacenada sin desajuste por error de tren de bits detectado. La variable picIdentifier indica el valor del PN (número de imagen) de la imagen almacenada (cuando la imagen almacenada es de corta duración) o del LPIN (número de imagen almacenada de larga duración) (cuando la imagen almacenada es de larga duración).

    – Si la imagen es de corta duración, el valor de picIdentifier será menor que MaxPN, que es el valor máximo posible del PN más 1.

  - De lo contrario (la imagen es de larga duración), el valor de picIdentifier será menor que MaxLPIN, que es valor máximo posible de LPIN más 1.

– Por otra parte, si payloadType es igual a 1, el bit 12 de ref_pic_id será igual a 0, y picIdentifier indicará el valor del PN de una imagen. El valor de PN será inferior a MaxPN que es igual al valor máximo posible de PN más 1. Todas las imágenes, en orden de decodificación y en la misma capa básica o capa de mejora, que comienzan con una imagen cuyo PN es igual a ref_pic_id & 0x0FFF, hasta una imagen (incluida) con PN igual a ((ref_pic_id & 0x0FFF) + delta_ref_pic_id) % MaxPN, cuando esas imágenes se encuentran en el tren de bits de vídeo, se indica que han sido total o parcialmente perdidas. MaxPN es igual al valor máximo posible de PN más 1.

– Por otra parte, si payloadType es igual a 2, el bit 12 de ref_pic_id será igual a 0, y picIdentifier indica el valor del PN de una imagen. El valor del PN será inferior a MaxPN que es igual al valor máximo posible del PN más 1. La imagen con TR igual a picIdentifier y, cuando el bit 13 de ref_pic_id es igual a 1, con ELNUM igual a ref_pic_id & 0x03C000, ha sido parcialmente perdida. Cada bloque es un macrobloque. El valor de data_partition_idc indica que todos los datos (cuando data_partition_idc es igual a 0), partición de datos de encabezamiento (cuando data_partition_idc es igual a 1), partición del vector de movimiento (cuando data_partition_idc es igual a 2) o partición de datos de coeficiente (cuando data_partition_idc es igual a 3) de los macrobloques identificados se han perdido. Todos los demás valores de data_partition_idc se reservan para uso futuro por el UIT-T y serán ignorados.

– Si (payloadType es igual a 5), no hay más especificaciones.

7.3 Utilización especifica de los mensajes H.264

Las características H.264 soportadas por la presente sintaxis incluyen lo siguiente:

– Imágenes de referencia de larga duración.

– Partición de datos.

– Conjunto de parámetros de imagen y secuencia.

La presente sintaxis no soporta las siguientes características H.264:

– Cuadros codificados utilizando codificación de cuadro-campo adaptativa por macrobloque.

– Imágenes de campo.

Cuando hay ref_pic_id o good_ref_pic_id[i], se aplica lo siguiente. Los 16 LSB indican el valor de la variable picIdentifier, que es igual a ref_pic_id & 0xFFFF o good_ref_pic_id[i] & 0xFFFF. Cuando payloadType es igual a 0, el bit 16 indica si la imagen de referencia es de larga duración (cuando el bit es igual a 1) o una imagen de referencia de corta duración (cuando el bit es igual 0). Los bits restantes de ref_pic_id o good_ref_pic_id[i] se reservan para un futuro uso por el UIT-T, serán iguales a 0 y serán ignorados.

Según el valor de payloadType se aplica lo siguiente:

– Si payloadType es igual a 0, ref_pic_id o good_ref_pic_id[i] identifica una imagen de referencia sin desajuste por error de tren de bit detectado. La variable ipicIdentifier indica el valor del FrameNum de la imagen de referencia (cuando la imagen de referencia es una imagen de corta duración) del LongTermFrameIdx (cuando la imagen de referencia es de larga duración).

  - Si la imagen es una imagen de referencia de corta duración, el valor de picIdentifier será inferior a MaxFrameNum.
  - Por otra parte (la imagen es una imagen de referencia de larga duración), el valor de picIdentifier no será superior a MaxLongTermFrameIdx.

– Por otra parte, si payloadType es igual a 1, el bit 16 será igual a 0 y la variable picIdentifier indica el valor de FrameNum de una imagen de referencia. El valor de FrameNum será inferior a MaxFrameNum. Todas las imágenes, en orden de decodificación, que comienzan con una imagen cuyo FrameNum es igual a ref_pic_id & 0x0FFFF, hasta una imagen (incluida) con FrameNum igual a ((ref_pic_id & 0x0FFFF) + delta_ref_pic_id) % MaxFrameNum, cuando esas imágenes se encuentran en el tren de bits de vídeo, se indica que han sido total o parcialmente perdidas.

– Por otra parte, si payloadType es igual a 2, el bit 16 será igual a 0 y la variable picIdentifier indica el valor de FrameNum de una imagen de referencia parcialmente perdida. El valor de FrameNum será inferior a MaxFrameNum. Cada bloque es un macrobloque. El valor de data_partition_idc indica que todos los datos (cuando data_partition_idc es igual a 0), partición de datos A (cuando data_partition_idc es igual a 1), partición de datos B (cuando data_partition_idc es igual a 2) o partición de datos C (cuando data_partition_idc es igual a 3) de los macrobloques identificados, se han perdido. Todos los demás valores de data_partition_idc se reservan para uso futuro por el UIT-T y serán ignorados.

– Por otra parte, si payloadType es igual a 3 ó 4, los 16 LSB de ref_pic_id indican el FrameNum que es igual a ref_pic_id & 0xFFFF, de una imagen de referencia para la cual hay un CRC para uno o todos los conjuntos de parámetros de secuencia o conjuntos de parámetros de imagen asociados. El valor de ref_pic_id & 0xFFFF será inferior a MaxFrameNum. Los bits restantes de ref_pic_id se reservan para una futura utilización por el UIT-T, serán iguales a 0 y serán ignorados. Un conjunto de parámetros con param_set_type igual a 0 es una unidad NAL de conjunto de parámetros secuencia. Un conjunto de parámetros con param_set_type igual a 1 es una unidad NAL de conjunto de parámetros imagen. Cuando se utilizan en el cálculo de un CRC, forbidden_zero_bit y nal_ref_idc de una unidad NAL de conjunto de parámetros imagen o secuencia se ponen a 0 y 3 respectivamente.

– Si (payloadType es igual a 5), no hay más especificaciones.

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	**Sistemas audiovisuales y multimedia**
Serie I	Red digital de servicios integrados
Serie J	Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedia
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	Gestión de las telecomunicaciones, incluida la RGT y el mantenimiento de redes
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos, comunicaciones de sistemas abiertos y seguridad
Serie Y	Infraestructura mundial de la información, aspectos del protocolo Internet y Redes de la próxima generación
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Impreso en Suiza
Ginebra, 2006