The Global Language of Business

Estándar de Datos de Etiquetas EPC

define el Código Electrónico de Producto (EPC™) y especifica el contenido de la memoria de las etiquetas RFID Gen 2

Versión 1.12, ratificada, mayo de 2019

Resumen del documento

Elemento del documento	Valor actual
Nombre del documento	Estándar de Datos de Etiquetas EPC
Fecha del documento	Mayo de 2019
Versión del documento	1.12
Emisión del documento	
Estado del documento	Ratificado
Descripción del documento	define el Código Electrónico de Producto (EPC™) y especifica el contenido de la memoria de las etiquetas RFID Gen 2

Colaboradores de la versión actual

Nombre	Organización	Función
Craig Alan Repec	GS1 Oficina Global	Editor
Mark Harrison	GS1 Oficina Global	Coeditor
Danny Haak	Nedap N.V.	Colaborador
Daniel Mullen	GS1 Oficina Global	Colaborador
Hemant Sahgal	Iris Software	Colaborador
Ralph Troger	GS1 Alemania	Colaborador

Registro de cambios

Versión	Fecha del cambio	Cambio hecho por	Resumen de cambios
1.9.1	8 de julio de 2015	D. Buckley	Se aplicó la nueva marca GS1
1.10	Marzo de 2017	Craig Alan Repec	Se detalló por completo en el Resumen a continuación
1.11	Sept. de 2017	Craig Alan Repec	Se detalló por completo en el Resumen a continuación
1.12	Abril de 2019	Craig Alan Repec y Mark Harrison	WR 19-076 Se agregó el URI de EPC para el UPUI, a fin de tener soporte para EU 2018/574, además del URI de EPC para el PGLN, el GLN de la parte AI (417), conforme a las Especificaciones Generales de GS1 19.1; Se agregaron las especificaciones normativas con respecto al manejo de la longitud del GCP para las llaves GS1 asignadas individualmente; Se corrigió la sintaxis del patrón de identidad pura de la ITIP; se introdujeron las selecciones de codificación y decodificación de “Entero de ancho fijo” para dar soporte a la codificación binaria de ITIP.

Exención de responsabilidad

Bajo su Política de PI, GS1® busca evitar la incertidumbre con respecto a las reclamaciones de propiedad intelectual solicitando a los participantes en el Grupo de Trabajo que elaboró este **Estándar de Datos de Etiquetas EPC** que acepten otorgar a los miembros de GS1 una licencia libre de regalías o una licencia RAND para Reclamaciones necesarias, tal como se define ese término en la Política de PI de GS1. Además, se llama la atención sobre la posibilidad de que la implementación de una o más características de esta Especificación pueda ser objeto de una patente u otro derecho de propiedad intelectual que no implique una reclamación necesaria. Ninguna patente ni cualquier otro derecho de propiedad intelectual está sujeto a las obligaciones de licencia de GS1. Además, el acuerdo para otorgar licencias provisto de conformidad con la Política de PI de GS1 no incluye derechos de propiedad intelectual ni reclamos de terceros que no hayan participado en el Grupo de Trabajo.

En consecuencia, GS1 recomienda que cualquier organización que desarrolle una implementación diseñada para guardar conformidad con esta Especificación determine si hay alguna patente que pueda abarcar una implementación específica que la organización esté desarrollando de conformidad con la Especificación y si es necesaria una licencia bajo una patente u otro derecho de propiedad intelectual. Esa determinación de la necesidad de una licencia debe hacerse teniendo en cuenta los detalles del sistema específico diseñado por la organización en consulta con su propio asesor de patentes.

ESTE DOCUMENTO SE PROPORCIONA "TAL CUAL" SIN GARANTÍA DE NINGÚN TIPO, INCLUYENDO CUALQUIER GARANTÍA DE COMERCIABILIDAD, NO INFRACCIÓN, APTITUD PARA UN PROPÓSITO PARTICULAR O CUALQUIER OTRA GARANTÍA QUE SURJA DE ESTA ESPECIFICACIÓN. GS1 se exime de toda responsabilidad por cualquier daño que surja del uso o mal uso de este Estándar, ya sean daños especiales, indirectos, consecuentes o compensatorios, incluida la responsabilidad por la infracción de cualquier derecho de propiedad intelectual en relación con el uso de información en este documento o la confianza en él.

GS1 se reserva el derecho de realizar cambios en este documento en cualquier momento y sin previo aviso. GS1 no ofrece ninguna garantía por el uso de este documento y no asume ninguna responsabilidad por los errores que puedan aparecer en el documento, ni se compromete a actualizar la información aquí contenida.

Índice

Prólogo

Resumen

El Estándar de Datos de Etiquetas EPC define el código electrónico de producto (EPC™) y especifica el contenido de la memoria de las etiquetas RFID Gen 2. En términos más detallados, el Estándar de Datos de Etiquetas cubre dos áreas amplias:

- La especificación del Código Electrónico de Producto, incluida su representación en diversos niveles de la arquitectura de EPCglobal y su correspondencia con las llaves GS1 y otros códigos existentes.
- La especificación de datos en las etiquetas RFID Gen 2, incluido el EPC, los datos de "memoria del usuario", la información de control y la información de fabricación de etiquetas.

Público al que se dirige este documento

El público objetivo de esta especificación incluye:

- Proveedores de middleware de EPC
- Usuarios y codificadores de etiquetas RFID
- Proveedores de lectores
- Desarrolladores de aplicaciones
- Integradores del sistema

Diferencias con el Estándar de Datos de Etiquetas EPC versión 1.6

El Estándar de Datos de Etiquetas EPC versión 1.7 es completamente compatible con la versión 1.6.

El Estándar de Datos de Etiquetas EPC versión 1.7 incluye las siguientes características nuevas o mejoradas:

- Se ha agregado un esquema de EPC nuevo, el esquema de identificador de partes y componentes (CPI);
- Se han corregido diversos errores tipográficos.

Diferencias con el Estándar de Datos de Etiquetas EPC versión 1.7

El Estándar de Datos de Etiquetas EPC versión 1.8 es completamente compatible con la versión 1.7.

El Estándar de Datos de Etiquetas EPC versión 1.8 incluye las siguientes mejoras:

- El Esquema de EPC GIAI ha recibido un valor de filtro adicional, "Vehículo ferroviario".

Diferencias con el Estándar de Datos de Etiquetas EPC versión 1.8

El Estándar de Datos de Etiquetas EPC versión 1.9 es completamente compatible con la versión 1.8.

El Estándar de Datos de Etiquetas EPC versión 1.9 incluye las siguientes mejoras:

- Se ha agregado un nuevo URI de clase de EPC para que represente la combinación de un GTIN más un lote (LGTIN).
- Se ha agregado un esquema de EPC nuevo, el Número de Cupón Global Serializado (SGCN), junto con la codificación binaria de SGCN-96.
- Se ha agregado un esquema de EPC nuevo, el Número Global de Relación del Servicio - Proveedor" (GSRNP), junto con la codificación binaria de GSRNP-96. Esto corresponde a la adición del AI (8017) a [GS1GS14.0];

- El Esquema de EPC GSRN existente recibe el título nuevo de Número Global de Relación del Servicio - Receptor con el fin de armonizar con la actualización [GS1GS14.0] al AI (8018). No obstante, el nombre del esquema de EPC y el URI no tienen cambios para conservar la compatibilidad con el TDS 1.8 y anteriores.
- Se agregaron AI nuevos a la Tabla de ID de objetos empaquetados para la memoria del usuario del EPC, a fin de armonizar el TDS con [GS1GS14.0], con lo que se garantiza que todos los AI se puedan codificar en portadores de datos tanto de códigos de barras como RFID.
  - Número de Componente de Empaque: AI (243)
  - Número Global de Cupón: AI (255)
  - Subdivisión de País de Origen: AI (427)
  - Número de Reembolso del Sector de Salud Nacional (NHRN) – Alemania PZN: AI (710)
  - Número de Reembolso del Sector de Salud Nacional (NHRN) – Francia CIP: AI (711)
  - Número de Reembolso del Sector de Salud Nacional (NHRN) – España CN: AI (712)
  - Número de Reembolso del Sector de Salud Nacional (NHRN) – Brasil DRN: AI (713)
  - Identificador de Partes y Componentes (8010)
  - Número de Serie del Identificador de Componente / Pieza (8011)
  - Número Global de Relación del Servicio - Prestador: AI (8017)
  - Número de Instancia de Relación de Servicios (SRIN): AI (8019)
  - URL Extendida de Empaque: AI (8200)
- Queda OBSOLETO "Datos secundarios para productos específicos de la industria de la salud" AI (22) en la Tabla de ID de objetos empaquetados para la memoria del usuario del EPC, a fin de armonizar el TDS con las Especificaciones Generales de GS1;
- Una nueva codificación binaria de EPC para el Identificador global de tipo de documento, GDTI-174, aceptará todos los valores del número de serie de GDTI que permite [GS1GS14.0] (1 - 17 caracteres alfanuméricos, en comparación con los caracteres numéricos 1 - 17 de las versiones anteriores de las Especificaciones Generales de GS1).
- Queda OBSOLETA la codificación binaria de EPC GDTI-113; en su lugar, se debe usar la codificación binaria GDTI-174
- Se actualizaron todas las referencias de versión y sección de [GS1GS14.0];
- Se marcó que la información de bits de atributo solo pertenece a las etiquetas Gen2 v 1.x;
- Se cambió **"*ItemReference*"** por **"*ItemRefAndIndicator*"** en la sintaxis general de SGTIN;
- Se corrigió la disposición sobre la cantidad de caracteres en la prueba de validación del método de codificación de "Cadena" de "menor que b/7" a "igual o menor que b/7";
- Se corrigieron varias erratas.

Diferencias con el Estándar de Datos de Etiquetas EPC versión 1.9

El Estándar de Datos de Etiquetas EPC versión 1.10 es completamente compatible con la versión 1.9.

El Estándar de Datos de Etiquetas EPC versión 1.10 incluye las siguientes mejoras:

- Se han agregado nuevos URI de EPC para representar los siguientes identificadores:
  - GINC
  - GS1N
  - Código de contenedor de BIC
- Se ha agregado una aclaración con respecto a los valores de filtro de SGTIN "Caja completa para transporte" y "Carga unitaria";
- El Esquema de EPC GDTI ha recibido un valor de filtro adicional, "Documento de viaje";

- El esquema de EPC ADI ha recibido una serie de valores de filtro adicionales, a fin de armonizar con la versión 2015 de la Especificación de la ATA 2000;
- Se han agregado AI nuevos a la Tabla de ID de objetos empaquetados para la memoria del usuario del EPC, a fin de armonizar el TDS con [GS1GS17.0], con lo que se garantiza que todos los AI se puedan codificar en portadores de datos tanto de códigos de barras como RFID.
  - Fecha límite de venta: AI (16)
  - Porcentaje de descuento de un cupón: AI (394n)
  - Zona de captura: AI (7005)
  - Fecha de primera congelación: AI (7006)
  - Fecha de cultivo: AI (7007)
  - Especies con fines de pesca: AI (7008)
  - Tipo de equipo de pesca: AI (7009)
  - Método de producción: AI (7010)
  - Versión de software: AI (8012)
  - Puntos de lealtad de un cupón: AI (8111)
- El AI (8102) de "Código extendido del cupón GS1-128 - NSC" se ha marcado como OBSOLETO;
- La cadena de formato para el AI (8007) de "Número de cuenta bancaria internacional (IBAN)" se ha corregido;
- La Tabla de codificación SGCN se ha corregido para incluir el encabezado SGCN;
- La identificación de etiqueta corta dentro del banco de memoria TID se ha actualizado para alinearse con [UHFC1G2v2.0];
- La correspondencia entre los EPC y las llaves GS1 se ha actualizado para aceptar los GCP de 4 y 5 dígitos, con el fin de alinearse con [GS1GS17.0];
- Resumen, Público y descripción general de las Diferencias se han pasado a una nueva sección de "Prólogo" después del Índice.

Diferencias con el Estándar de Datos de Etiquetas (TDS) EPC, versión 1.10

TDS v 1.11 es totalmente compatible con versiones anteriores de TDS v 1.10.

TDS v 1.11 incluye las siguientes mejoras:

- Se ha agregado un nuevo esquema de EPC, el Artículo Comercial Individual (ITIP), junto con las codificaciones binarias ITIP-110 e ITIP-212.
- Se han agregado los siguientes AI nuevos a la Tabla de ID de objetos empaquetados para la memoria del usuario del EPC, a fin de armonizar el TDS con [GS1GS17.1], con lo que se garantiza que todos los AI se puedan codificar en portadores de datos tanto de códigos de barras como RFID:
  - GLN del lugar de producción o de servicio: AI (416)
  - ID de lote de renovación: AI (7020)
  - Estado funcional: AI (7021)
  - Estado de revisión: AI (7022)
  - Identificador Global Individual de Activo (GIAI) de un montaje: AI (7023)
- La cadena de formato para los AI 91-99 se ha revisado para permitir hasta 90 caracteres (anteriormente hasta 30), con el fin de armonizar el TDS con [GS1GS17.0];

  NOTA: Para armonizar con GenSpecs v 17.1, que ha extendido la longitud de los AI 91-99 a 90 (anteriormente 30) caracteres alfanuméricos, TDS v 1.11 ha extendido el formato de cadena de los AI 91-99 (codificado por medio de objetos empaquetados en la memoria del usuario) de 1*30an (alfanumérico, longitud de 1 a 30) a 1*an (alfanumérico, sin límite superior).

Esta revisión de las tablas F.1 y F.2 del TDS es totalmente compatible con versiones anteriores, lo que permite que una etiqueta escrita según el TDS 1.10 se decodifique correctamente según el TDS 1.11. También es en su mayoría compatible con versiones posteriores, lo que permite que una etiqueta escrita según el TDS 1.11 se decodifique correctamente según el TDS 1.10, siempre que la longitud de los AI 91, ..., 99 sea 30 o menos. Una etiqueta escrita según el TDS 1.10 con un valor más largo para uno de estos AI puede señalar un error que indica que el valor es demasiado largo, pero otros AI se decodificarán correctamente. Otro problema menor es que el algoritmo de codificación ya no impondrá un límite superior en la longitud de un valor codificado, por lo que será posible codificar un valor de carácter de AI de 91-99 que sea demasiado largo para GenSpecs (por ejemplo, 100 caracteres). Por lo tanto, **para garantizar el cumplimiento con GenSpecs y el resto del Sistema GS1, los valores de carácter de AI 91-99 codificados en la memoria del usuario no deben exceder los 90 caracteres de longitud**.

- Se marcaron todos los encabezados binarios de EPC previamente reservados para las codificaciones de 64 bits como ahora “Reservado para uso futuro” (RFU), lo que refleja el declive de julio del 2009 de las codificaciones de 64 bits.

Diferencias con el Estándar de Datos de Etiquetas (TDS) EPC, versión 1.11

TDS v 1.12 es totalmente compatible con versiones anteriores de TDS v 1.11.

TDS v 1.12 incluye las siguientes mejoras:

- Se ha agregado el siguiente esquema de EPC:
  - UPUI
  - PGLN
- Se ha agregado una guía (a la sección 7) para determinar la longitud del componente de Prefijo de empresa de EPC para las llaves GS1 asignadas de forma individual
- Se han agregado métodos de codificación y decodificación de “Entero de ancho fijo” (a la sección 14) en apoyo del ITIP,
- El método de codificación para los componentes Pieza y Total del ITIP se ha corregido de “Cadena” a “Entero de ancho fijo”.
- Se han agregado los siguientes AI nuevos a la Tabla de ID de objetos empaquetados para la memoria del usuario del EPC, a fin de armonizar el TDS con [GS1GS19.1], con lo que se garantiza que todos los AI se puedan codificar en portadores de datos tanto de códigos de barras como RFID:
  - Variante de producto de consumo: AI (22)
  - Extensión serializada de GTIN (TPX) controlada por terceros: AI (235)
  - Número Global de Localización de la Parte: AI (417)
  - Número de Reembolso del Sector de Salud Nacional (NHRN) – Portugal AIM AI (714)
  - GS1 UIC con extensión 1 e índice de importador (conforme a UE 2018/574) AI (7040)
  - Número global de modelo: AI (8013)
  - Identificación de las partes de un artículo comercial (ITIP) contenidas en una unidad logística: AI (8026)
  - Identificación del código del cupón para uso en Norteamérica: AI (8112)

Estado de este documento

En esta sección se describe el estado de este documento al momento de su publicación. Otros documentos pueden reemplazar este documento. El estado más reciente de esta serie de documentos se mantiene en GS1. Consulte ***http://www.gs1.org/standards*** para ver más información.

Esta versión del Estándar de Datos de Etiquetas EPC 1.12 se ha ratificado y ha completado todos los demás pasos de GSMP, incluida la revisión de PI.

1 Introducción

El Estándar de Datos de Etiquetas EPC define el Código Electrónico de Producto (EPC™) y especifica el contenido de la memoria de las etiquetas RFID Gen 2. En términos más detallados, el Estándar de Datos de Etiquetas cubre dos áreas amplias:

- La especificación del Código Electrónico de Producto, incluida su representación en diversos niveles de la arquitectura de EPCglobal y su correspondencia con las llaves GS1 y otros códigos existentes.
- La especificación de datos en las etiquetas RFID Gen 2, incluido el EPC, los datos de "memoria del usuario", la información de control y la información de fabricación de etiqueta.

El Código Electrónico de Producto es un identificador universal para cualquier objeto físico. Se usa en sistemas de información que requieren rastrear o hacer referencia de alguna otra manera a objetos físicos. Un subconjunto muy grande de aplicaciones que usan el Código Electrónico de Producto también depende de las etiquetas RFID como portador de datos. Por este motivo, una gran parte del Estándar de Datos de Etiquetas aborda la codificación de los Códigos Electrónicos de Producto en las etiquetas RFID, junto con la definición de los estándares para otros datos además del EPC que se pueden almacenar en una etiqueta RFID Gen 2.

Por lo tanto, las dos áreas claras cubiertas por el Estándar de Datos de Etiquetas (el EPC y RFID) se superponen en las partes en que se analiza la codificación del EPC en las etiquetas RFID. No obstante, siempre debe recordarse que EPC y RFID no son sinónimos en lo más mínimo: EPC es un identificador, mientras que RFID es un portador de datos. Las etiquetas RFID contienen otros datos además de identificadores de EPC (y en algunas aplicaciones pueden no tener ningún identificador de EPC en lo absoluto), y el identificador de EPC existe en contextos fuera de RFID (estos contextos distintos de RFID incluyen la forma de URI usada dentro de los sistemas de información, los URI de EPC impresos legibles para el ser humano y los identificadores de EPC derivados de datos de códigos de barras tras los procedimientos en este estándar).

2 Terminología y convenciones tipográficas

Dentro de esta especificación, los términos DEBE, NO DEBE, DEBERÍA, NO DEBERÍA, PUEDE, NO NECESITA, PUEDE y NO PUEDE deben interpretarse como se especifica en el Anexo ***G*** de las Directivas ISO/IEC, Parte 2, 2001, 4.ª edición [ISODir2]. Cuando se usan de esta manera, estos términos siempre se mostrarán EN MAYÚSCULAS; cuando estas palabras aparecen en una tipografía normal, se pretende que tengan su significado habitual.

Todas las secciones de este documento, con la excepción de la Sección ***1***, son normativas, excepto cuando se indique explícitamente que no son normativas.

Las siguientes convenciones tipográficas se utilizan en todo el documento:

- El tipo EN MAYÚSCULAS se utiliza para los términos especiales de [ISODir2] indicados anteriormente.
- El tipo monoespacio se utiliza para ilustraciones de identificadores y otras cadenas de caracteres que existen dentro de los sistemas de información.
- ➢ Los marcadores de posición para los cambios que deben realizarse en este documento antes de que llegue a la etapa final de la especificación de EPCglobal aprobada están precedidos por una punta de flecha que apunta hacia la derecha, como en este párrafo.

El término "etiqueta RFID Gen 2" (o simplemente "etiqueta Gen 2") como se usa en esta especificación se refiere a cualquier etiqueta RFID que se ajuste a la interfaz aire UHF Clase 1 Generación 2 de EPCglobal, versión 1.2.0 o posterior [UHFC1G2], así como a cualquier etiqueta RFID que se ajuste a otro estándar de interfaz aire que comparte el mismo mapa de memoria. En lo que se refiere a los bits, las direcciones dentro de los bancos de memorias de etiqueta Gen 2 se indican usando numerales hexadecimales que termina con un subíndice "h"; por ejemplo, 20h denota dirección de bits hexadecimal 20 (32 decimal).

3 Descripción general del Estándar de Datos de Etiquetas

Esta sección proporciona una descripción general del Estándar de Datos de Etiquetas y cómo embonan las partes.

El Estándar de Datos de Etiquetas cubre dos áreas amplias:

- La especificación del código EPC, incluida su representación en diversos niveles de la arquitectura de EPCglobal y su correspondencia con las llaves GS1 y otros códigos existentes.

- La especificación de datos en las etiquetas RFID Gen 2, incluido el EPC, los datos de "memoria del usuario", la información de control y la información de fabricación de etiquetas.

El Código Electrónico de Producto es un identificador universal para cualquier objeto físico. Se usa en sistemas de información que requieren rastrear o hacer referencia de alguna otra manera a objetos físicos. En los sistemas informáticos, incluidos los documentos electrónicos, las bases de datos y los mensajes electrónicos, el EPC toma la forma de un identificador de recursos uniforme (URI) de Internet. Esto es un hecho independientemente de si el EPC originalmente se leía en una etiqueta RFID o algún otro tipo de portador de datos. Este URI se llama "URI de identidad pura EPC". El siguiente es un ejemplo de URI de identidad pura EPC:

```
urn:epc:id:sgtin:0614141.112345.400
```

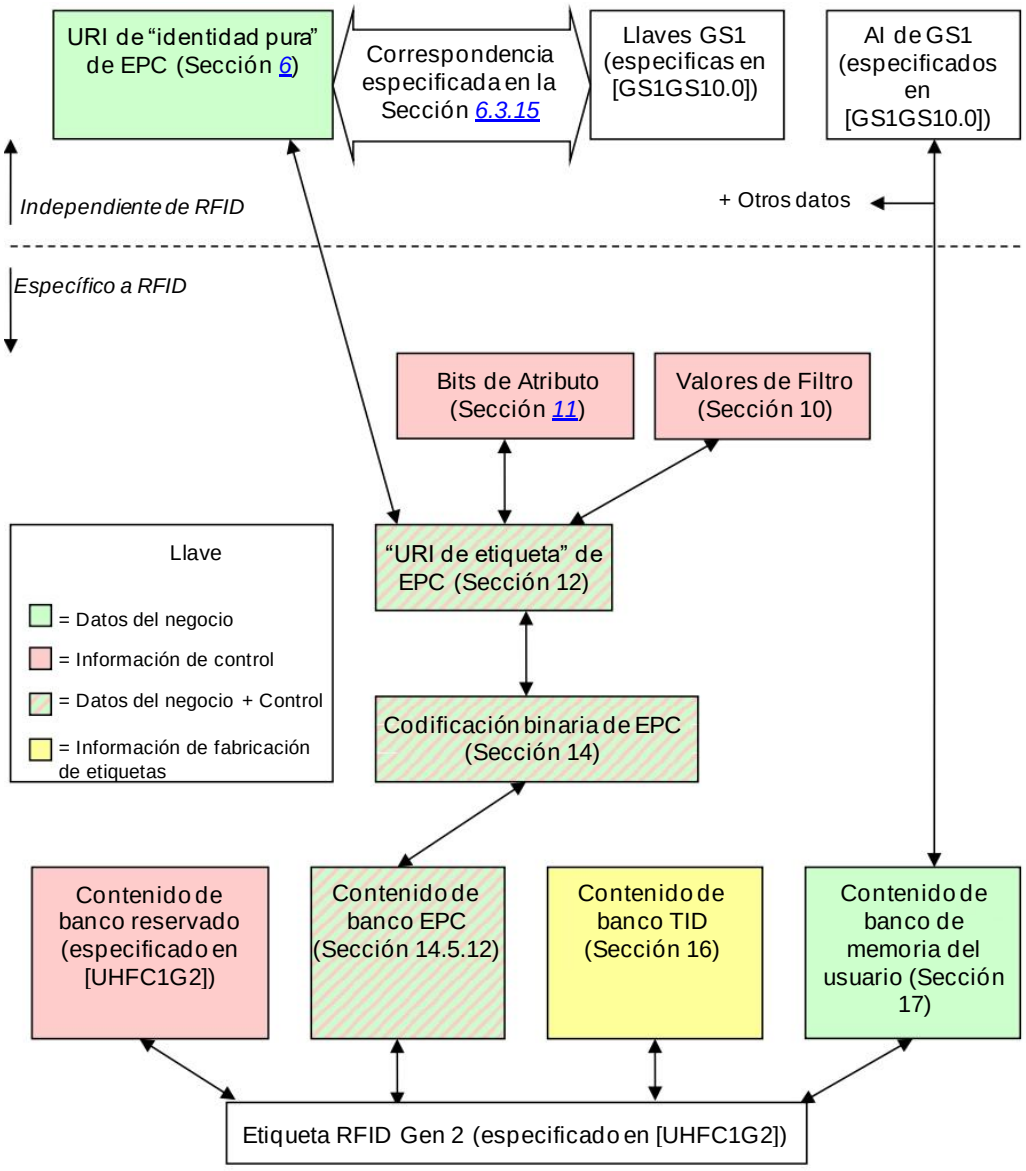
Un subconjunto muy grande de aplicaciones que usan el Código Electrónico de Producto también depende de las etiquetas RFID como portador de datos. El RFID con frecuencia es una tecnología de portador de datos muy adecuada para aplicaciones que implican visibilidad de objetos físicos. Esto se debe a que RFID permite que los datos se adjunten físicamente a un objeto de manera tal que la lectura de los datos sea mínimamente invasiva para los procesos de manejo de materiales. Por este motivo, una gran parte del Estándar de Datos de Etiquetas aborda la codificación de los códigos EPC en las etiquetas RFID, junto con la definición de los estándares para otros datos además del EPC que se pueden almacenar en una etiqueta RFID Gen 2. Debido a las limitaciones de memoria de las etiquetas RFID, el EPC no se almacena en forma de URI en la etiqueta, en lugar de ello se codifica en una representación binaria compacta. A esto se le denomina "codificación binaria de EPC".

Por lo tanto, las dos áreas claras cubiertas por el Estándar de Datos de Etiquetas (el EPC y RFID) se superponen en las partes en que se analiza la codificación del EPC en las etiquetas RFID. No obstante, siempre debe recordarse que EPC y RFID no son sinónimos en lo más mínimo: EPC es un identificador, mientras que RFID es un portador de datos. Las etiquetas RFID contienen otros datos además de identificadores de EPC (y en algunas aplicaciones pueden no tener ningún identificador de EPC en lo absoluto), y el identificador de EPC existe en contextos fuera de RFID (estos contextos distintos de RFID incluyen actualmente la forma de URI usada dentro de los sistemas de información, los URI de EPC impresos legibles para el ser humano y los identificadores de EPC derivados de datos de códigos de barras tras los procedimientos en este estándar).

El término "Código Electrónico de Producto" (o "EPC") se usa para referirse al EPC independientemente de la forma concreta usada para representarlo. El término "URI de identidad pura EPC" se usa específicamente para referirse a la forma de texto que el EPC toma en los sistemas informados, incluyendo los documentos electrónicos, las bases de datos y los mensajes electrónicos. El término "codificación binaria de EPC" se usa específicamente para referirse a la forma que toma el EPC en la memoria de las etiquetas RFID.

El siguiente diagrama ilustra las partes del Estándar de Datos de Etiquetas y la manera en que embonan. (Los colores en el diagrama se refieren a los tipos de datos que pueden usarse en las etiquetas RFID, explicadas a mayor detalle en la Sección *9.1.)*

Figura 3-1 Organización del Estándar de Datos de Etiquetas EPC



Las primeras secciones definen los aspectos del Código Electrónico de Producto independientes de RFID.

En la Sección 4 se incluye una descripción general del Código Electrónico de Producto (EPC) y cómo se relaciona con otros estándares de GS1 y las Especificaciones Generales de GS1.

En la Sección 6 se especifica la forma de URI de identidad pura EPC del EPC. Esta es una forma textual del EPC y se recomienda su uso en aplicaciones y documentos comerciales como identificador universal para todo objeto físico en el que se mantiene su información de visibilidad. En particular, esta forma es lo que se usa como la dimensión "qué" de los datos de visibilidad en la especificación de los Servicios de Información de EPC (EPCIS) y también está disponible como salida de la interfaz de Eventos de Nivel de Aplicación (ALE).

En la Sección 7 se especifica la correspondencia entre los URI de identidad pura EPC según se definen en la Sección 6 y las cadenas de elementos de código de barras definidas en las Especificaciones generales de GS1.

La sección 7.9 especifica el URI del patrón de identidad pura, que es una sintaxis para representar conjuntos de EPC relacionados, como todos los EPC de un artículo comercial determinado, independientemente del número de serie.

Las secciones restantes abordan temas que son específicos a la RFID, incluidas las formas específicas de RFID del EPC, así como otros datos además del EPC que pueden almacenarse en etiquetas RFID Gen 2.

En la Sección 9 se incluye información general sobre la estructura de la memoria de las etiquetas RFID Gen 2.

En las Secciones 10 y 11 se especifica la información de "control" que se almacena en el banco de memoria EPC de las etiquetas Gen 2 junto con una forma codificada binaria del EPC (Codificación Binaria de EPC). Las aplicaciones de captura de datos por RFID utilizan la información de control para dirigir el proceso de captura de datos, mediante la entrega de pistas sobre el tipo de objeto en el que se ha fijado la etiqueta. La información de control no es parte del EPC y comprende cualquier parte de la identidad única de un objeto etiquetado. Hay dos tipos de información de control especificada: el "valor de filtro" (Sección 10) que facilita la lectura de las etiquetas deseadas en un entorno donde puede haber otras etiquetas, como leer una etiqueta de una tarima en presencia de una gran cantidad de etiquetas a nivel de artículo y "bits de atributo" (Sección 11) que brinda información adicional sobre atributos especiales, como alertar sobre la presencia de material peligroso. Los mismos "bits de atributo" están disponibles independientemente del tipo de EPC que se utilice, mientras que los "valores de filtro" disponibles son diferentes según el tipo de EPC (y con determinados tipos de EPC no se dispone de ningún valor de filtro).

La sección 12 especifica los identificadores de recursos uniforme "etiqueta", que es una representación de cadena compacta para todo el contenido de datos del banco de memoria EPC de etiquetas RFID Gen 2. Este contenido de datos incluye el EPC junto con la información de "control" como se define en las Secciones 10 y 11 . En el URI "etiqueta", el contenido de EPC del banco de memoria EPC se representa en una forma similar al URI de identidad pura EPC. Sin embargo, a diferencia del URI de identidad pura EPC, el URI "etiqueta" también incluye el contenido de información de control del banco de memoria EPC. Se recomienda el uso de la forma de URI "etiqueta" en aplicaciones de captura que necesitan leer información de control para capturar datos correctamente, o que necesitan escribir el contenido completo del banco de memoria EPC. Los URI "etiqueta" se utilizan en la interfaz de Eventos de Nivel de Aplicación (ALE), como entrada (al escribir etiquetas) y como salida (al leer etiquetas).

La Sección 13 especifica el URI de patrón de etiqueta EPC, que es una sintaxis para representar conjuntos de etiquetas RFID relacionadas en función del contenido del EPC, como todas las etiquetas que contienen EPC para un determinado rango de números de serie para un artículo comercial determinado.

Las secciones 14 y 14.5.1.2 especifican el contenido del banco de memoria EPC de una etiqueta RFID Gen 2 a nivel de bit. En la Sección 14 se especifica cómo traducir entre el URI "etiqueta" y la codificación binaria de EPC. La codificación binaria es una representación a nivel de bits de lo que realmente está almacenado en la etiqueta y también es lo que se transporta a través de la interfaz del Protocolo de Lectura de Bajo Nivel (LLRP). En la Sección 14.5.1.2 se especifica cómo se combina esta codificación binaria con los bits de atributo y otra información de control en el banco de memoria EPC.

En la Sección 16 se especifica la codificación binaria del banco de memoria TID de las etiquetas RFID Gen 2.

En la Sección 17 se especifica la codificación binaria del banco de memoria del usuario de las etiquetas RFID Gen 2.

4 El Código Electrónico de Producto: un identificador universal para objetos físicos

El Código Electrónico de Producto está diseñado para facilitar los procesos y aplicaciones comerciales que necesitan manipular datos de visibilidad, datos sobre observaciones de objetos físicos. El EPC es un identificador universal que proporciona una identidad única para cualquier objeto físico. El EPC está diseñado para ser único en todos los objetos físicos del mundo, en todo momento y en todas las categorías de objetos físicos. Se ha diseñado específicamente para que lo utilicen aplicaciones comerciales que requieran hacer un seguimiento de todas las categorías de objetos físicos, cualesquiera que sean.

Por el contrario, con las llaves de identificación GS1 definidas en las Especificaciones Generales GS1 [GS1GS] se pueden identificar categorías de objetos (GTIN), objetos únicos (SSCC, GLN, GIAI, GSRN, CPID) o un híbrido (GRAI, GDTI, GCN) que puede identificar categorías u objetos únicos dependiendo de la ausencia o presencia de un número de serie. (Con otras dos llaves, GINC y GS1N, se identifican agrupaciones lógicas, no objetos físicos). El GTIN, como la única llave de identificación de categoría, requiere un número de serie separado para identificar un objeto de forma única, pero ese número de serie no se considera parte de la llave de identificación.

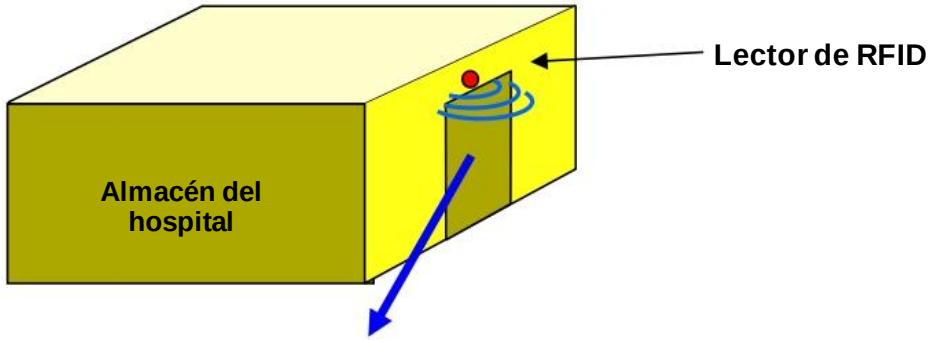
Existe una correspondencia bien definida entre los EPC y las llaves GS1. Esta correspondencia permite que todo objeto físico que ya se ha identificado con una llave GS1 (o una combinación de llave GS1 + número de serie) se use en un contexto de EPC en el que se pueda observar cualquier categoría de objeto físico. Del mismo modo, permite que los datos del EPC capturados en un contexto de amplia visibilidad se relacionen con otra información empresarial específica a la categoría del objeto en cuestión y que emplea llaves GS1.

En el resto de esta sección se profundiza en estos puntos.

4.1 Ejemplo de la necesidad de un identificador universal

El siguiente ejemplo ilustra cómo surgen los datos de visibilidad y el papel que juega el EPC como identificador único para cualquier objeto físico. En este ejemplo, hay un almacén en un hospital que contiene muestras radiactivas, entre otras cosas. El oficial de seguridad del hospital debe realizar un seguimiento de las cosas que han estado en el almacén y durante cuánto tiempo para asegurarse de que la exposición se mantenga dentro de límites aceptables. A cada objeto físico que llega a entrar en el almacén se le asigna un Código de Producto Electrónico único, que se codifica en una etiqueta RFID adherida al objeto. Un lector de RFID colocado en la puerta del almacén genera datos de visibilidad a medida que los objetos entran y salen del almacén, como se ilustra a continuación.

Figura 4-1 Ejemplo de flujo de datos de visibilidad



Flujo de datos de visibilidad en la entrada del almacén			
Hora	**Entrada/ Salida**	**EPC**	**Comentario**
8:23 a. m.	Entrada:	urn:epc:id:sgtin:0614141.012345.62852	Jeringa de 10cc # 62852 (artículo comercial)
8:52 a. m.	Entrada:	urn:epc:id:grai:0614141.54321.2528	Contenedor de farmacéuticos # 2528 (transporte reutilizable)
8:59 a. m.	Entrada:	urn:epc:id:sgtin:0614141.012345.1542	Jeringa de 10cc # 1542 (artículo comercial)
9:02 a. m.	Salida	urn:epc:id:giai:0614141.17320508	Bomba de infusión # 52 (activo fijo)
9:32 a. m.	Entrada:	urn:epc:id:gsrn:0614141.0000010253	Enfermera Jones (relación con el servicio)
9:42 a. m.	Salida	urn:epc:id:gsrn:0614141.0000010253	Enfermera Jones (relación con el servicio)
9:52 a. m.	Entrada:	urn:epc:id:gdti:0614141.00001.1618034	Historia clínica del paciente Smith (documento)

Como muestra la ilustración, el flujo de datos de interés para el oficial de seguridad es una serie de eventos, cada uno de los cuales identifica un objeto físico específico y el momento en que entró o salió del almacén. El EPC único para cada objeto es un identificador que se puede utilizar para impulsar el proceso empresarial. En este ejemplo, el EPC (en forma de URI de identidad pura EPC) sería una llave principal de una base de datos que rastrea la exposición acumulada para cada objeto físico; cada par de eventos de entrada/salida para un objeto dado se usaría para actualizar la base de datos de exposición acumulada.

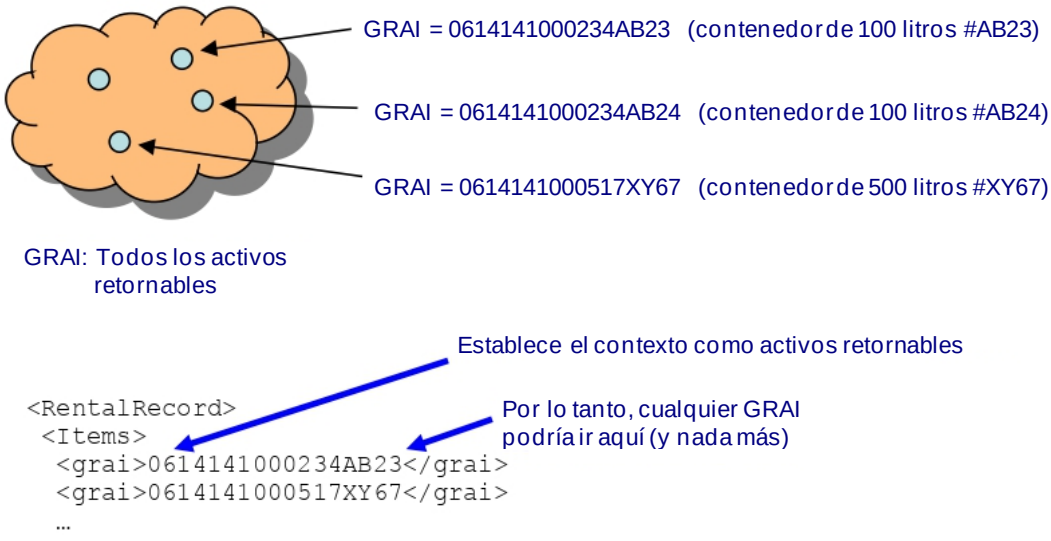
Este ejemplo ilustra cómo el EPC es un identificador ***universal*** único para cualquier objeto físico. Los artículos que se rastrean aquí incluyen todo tipo de cosas: artículos comerciales, transportes reutilizables, activos fijos, relaciones de servicio, documentos, entre otras cosas que puedan encontrarse. Al usar el EPC, la aplicación puede usar un solo identificador para referirse a algún objeto físico y no es necesario hacer un caso especial para cada categoría de cosas.

4.2 Uso de identificadores en un contexto de datos comerciales

En términos generales, un identificador es un miembro de un conjunto (o "espacio de nombres") de cadenas (nombres), de modo que cada identificador está asociado con una cosa o concepto específico en el mundo real. Los identificadores se utilizan dentro de los sistemas de información para referirse a la cosa o concepto del mundo real en cuestión. Un identificador puede encontrarse en un registro o archivo electrónico, en una base de datos, en un mensaje electrónico o en cualquier otro contexto de datos. En un contexto determinado, el productor y el consumidor deben acordar qué espacio de nombres de identificadores se utilizará; dentro de ese contexto, se puede utilizar cualquier identificador que pertenezca a ese espacio de nombres.

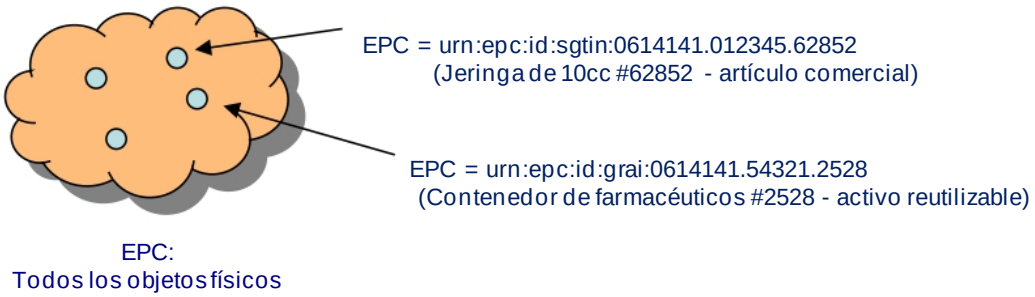
Las llaves definidas en las Especificaciones Generales GS1 [GS17.0] son cada una un espacio de nombres de identificadores para una categoría particular de una entidad del mundo real. Por ejemplo, el Identificador Global de Activos Retornables (GRAI) es una llave que se utiliza para identificar activos retornables, como contenedores de plástico y tarimas. El conjunto de códigos GRAI se puede considerar como identificadores para los miembros del conjunto "todos los activos retornables". Se puede utilizar un código GRAI en un contexto en el que solo se esperan activos retornables; por ejemplo, en un contrato de alquiler de una empresa de servicios de mudanzas que alquila cajas de plástico retornables a los clientes para que las empaqueten durante una mudanza. Esto se ilustra a continuación.

Figura 4-2 Ilustración del espacio de nombres GRAI



La parte superior de la figura ilustra el espacio de nombres del identificador GRAI. La parte inferior de la figura muestra cómo se podría utilizar un GRAI en el contexto de un contrato de alquiler donde solo se espera un GRAI.

Figura 4-3 Ilustración del espacio de nombres del EPC




```
<EPCISDocument>
 <ObjectEvent>
  <epcList>
   <epc>urn:epc:id:sgtin:0614141.012345.62852</epc>
   <epc>urn:epc:id:grai:0614141.54321.2528</epc>
   ...
```

Establece el contexto como todos los objetos físicos.

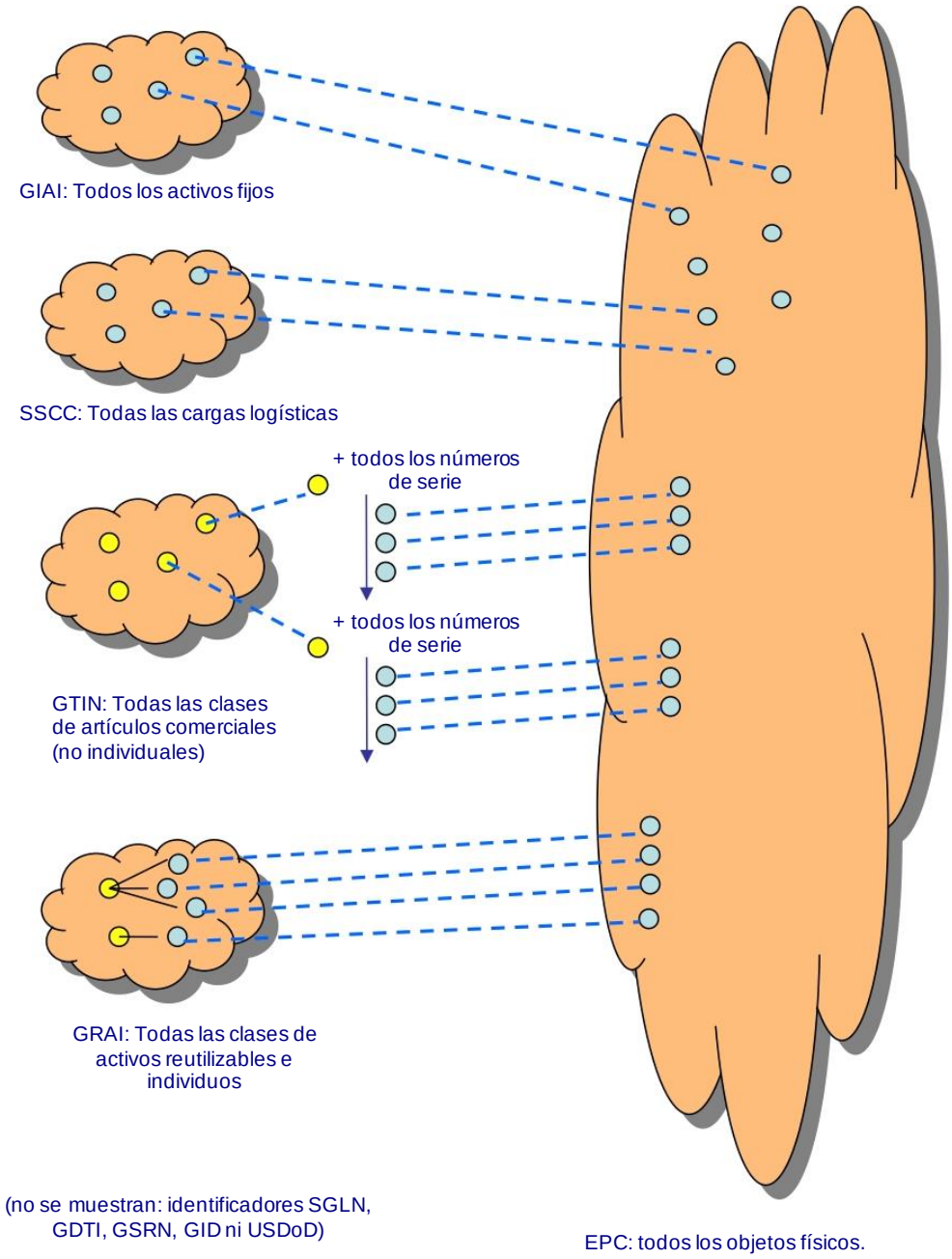
Por lo tanto, cualquier EPC podría ir aquí.

En contraste, el espacio de nombres del EPC es un espacio de identificadores para ***cualquier*** objeto físico. El conjunto de EPC se puede considerar como identificadores para los miembros del conjunto "todos los objetos físicos". Los EPC se utilizan en contextos donde puede aparecer cualquier tipo de objeto físico, como en la serie de observaciones que surgen en el ejemplo anterior del almacén del hospital. Tenga en cuenta que el URI de EPC ilustrado en la *Figura 4-3* incluye cadenas como sgtin, grai, etc., como parte del identificador URI de EPC. Esto difiere de las llaves GS1, en donde esa indicación no forma parte de la llave misma; en cambio, se indica fuera de la llave, como en el nombre del elemento XML <grai> en el ejemplo de la *Figura 4-2* en el identificador de aplicación (AI) que acompaña a una llave GS1 en una cadena de elementos GS1.

4.3 Relación entre los EPC y las llaves GS1

Existe una relación bien definida entre los EPC y las llaves GS1. Para cada llave GS1 que denota un objeto físico individual, hay un EPC correspondiente, que incluye un URI de EPC y una codificación binaria que se debe usar en las etiquetas RFID. Además, cada llave GS1 que denota una clase o agrupación de objetos físicos tiene una forma de URI correspondiente. Estas correspondencias se definen formalmente mediante las reglas de conversión especificadas en la Sección 7, que definen cómo asignar una llave GS1 al valor de EPC correspondiente y viceversa. La correspondencia bien definida entre las llaves GS1 y los EPC permite una migración fluida de datos entre los contextos de llaves GS1 y EPC según sea necesario.

Figura 4-4 Ilustración de la relación de los espacios de nombres de llaves GS1 e identificadores EPC.



No todas las llaves GS1 corresponden a un EPC, ni viceversa. Específicamente:

- Un Número Global de Artículo Comercial (GTIN) por sí solo no corresponde a un EPC, porque un GTIN identifica una ***clase*** de artículos comerciales, no un artículo comercial individual. Sin embargo, la combinación de un GTIN y un número de serie único ***corresponde*** a un EPC. Esta combinación se denomina Número Global de Artículo Comercial Serializado o SGTIN. Las Especificaciones Generales de GS1 no definen al SGTIN como una llave GS1.

- En las Especificaciones Generales GS1, el Identificador Global de Activos Retornables (GRAI) se puede usar para identificar una ***clase*** de activos retornables o un activo retornable individual, dependiendo de si se incluye el número de serie opcional. Solo la forma que incluye un número de serie y, por lo tanto, identifica un artículo individual, tiene un EPC correspondiente. Lo mismo ocurre con el Identificador Global de Tipo de Documento (GDTI) y el Número de Cupón Global (GCN); en lo sucesivo, en este contexto, "Número de Cupón Global Serializado (SGCN)".
- Hay un EPC correspondiente a cada Número Global de Localización (GLN) y también hay un EPC correspondiente a cada combinación de un GLN con un componente de extensión. En conjunto, estos EPC se denominan SGLN. [1]
- Los EPC incluyen identificadores para los que no existe una llave GS1 correspondiente. Estos incluyen el Identificador General y el identificador del Departamento de Defensa de los EE. UU.

En la siguiente tabla se resumen los esquemas de EPC definidos en esta especificación y su correspondencia con las llaves GS1.

Tabla 4-1 Esquemas de EPC y llaves GS1 correspondientes.

Esquema de EPC	Codificaciones de etiquetas	Llave GS1 correspondiente	Uso típico
sgtin	sgtin-96 sgtin-198	Llave GTIN (más el número de serie agregado)	Artículo comercial
sscc	sscc-96	SSCC	Carga de tarima u otra carga unitaria de logística
sgln	sgln-96 sgln-195	GLN de localización física (con o sin extensión adicional)	Localización
grai	grai-96 grai-170	GRAI (número de serie obligatorio)	Activo retornable/reutilizable
giai	giai-96 giai-202	GIAI	Activo fijo
gsrn	gsrn-96	GSRN - Receptor	Hospitalización o membresía en un club.
gsrnp	gsrnp-96	GSRN para el proveedor del servicio	Cuidador médico o club de lealtad
gdti	gdti-96 ***gdti-113*** (OBSOLETO) gdti-174	GDTI (número de serie obligatorio)	Documento
cpi	cpi-96 cpi-var	[ninguno]	Industrias técnicas (por ejemplo, automotriz): componentes y partes.
sgcn	sgcn-96	GCN (número de serie obligatorio)	Cupón
ginc	[ninguno]	GINC	Agrupación lógica de mercancías destinadas a transportarse en conjunto, asignada por un transportista
GS1n	[ninguno]	GS1N	Agrupación lógica de unidades logísticas que viajan bajo aviso de envío y/o acuse de recibo

[1] Tenga en cuenta que en este contexto, la letra "S" no significa "serializado" como en el SGTIN. Consulte la Sección *6.3.3* para obtener una explicación.

Esquema de EPC	Codificaciones de etiquetas	Llave GS1 correspondiente	Uso típico
itip	itip-110 itip-212	(8006) + (21)	Una de las múltiples partes que abarcan y están subordinadas a un todo (que, a su vez, se identifica mediante un SGTIN o la combinación de AI 01 + 21).
upui	[ninguno]	GTIN + TPX	Identificación del paquete para combatir el comercio ilícito
pgln	[ninguno]	GLN de la parte	Identificación del operador económico; identificación de la parte propietaria o poseedora en la Cadena de Custodia (CoC)/Cadena de Propiedad (CoO).
gid	gid-96	[ninguno]	Sin especificar
usdod	usdod-96	[ninguno]	Cadena de suministro del Departamento de Defensa de los EE. UU.
adi	adi-var	[ninguno]	Aeroespacial y defensa - aeronaves y otras partes y artículos
bic	[ninguno]	[ninguno]	Contenedores de transporte intermodal

4.4 Uso del EPC en el Marco de la Arquitectura de EPCglobal

El Marco de la Arquitectura de EPCglobal [EPCAF] es un conjunto de estándares de hardware, software y datos, junto con servicios de red compartidos que pueden ser operados por EPCglobal, sus delegados o proveedores externos en el mercado, todo al servicio del objetivo común de mejorar flujos comerciales y aplicaciones informáticas mediante el uso de Códigos Electrónicos de Productos (EPC). El Marco de la Arquitectura de EPCglobal incluye estándares de software en diversos niveles de abstracción, de interfaces de bajo nivel a dispositivos lectores de RFID, hasta el nivel de aplicaciones comerciales.

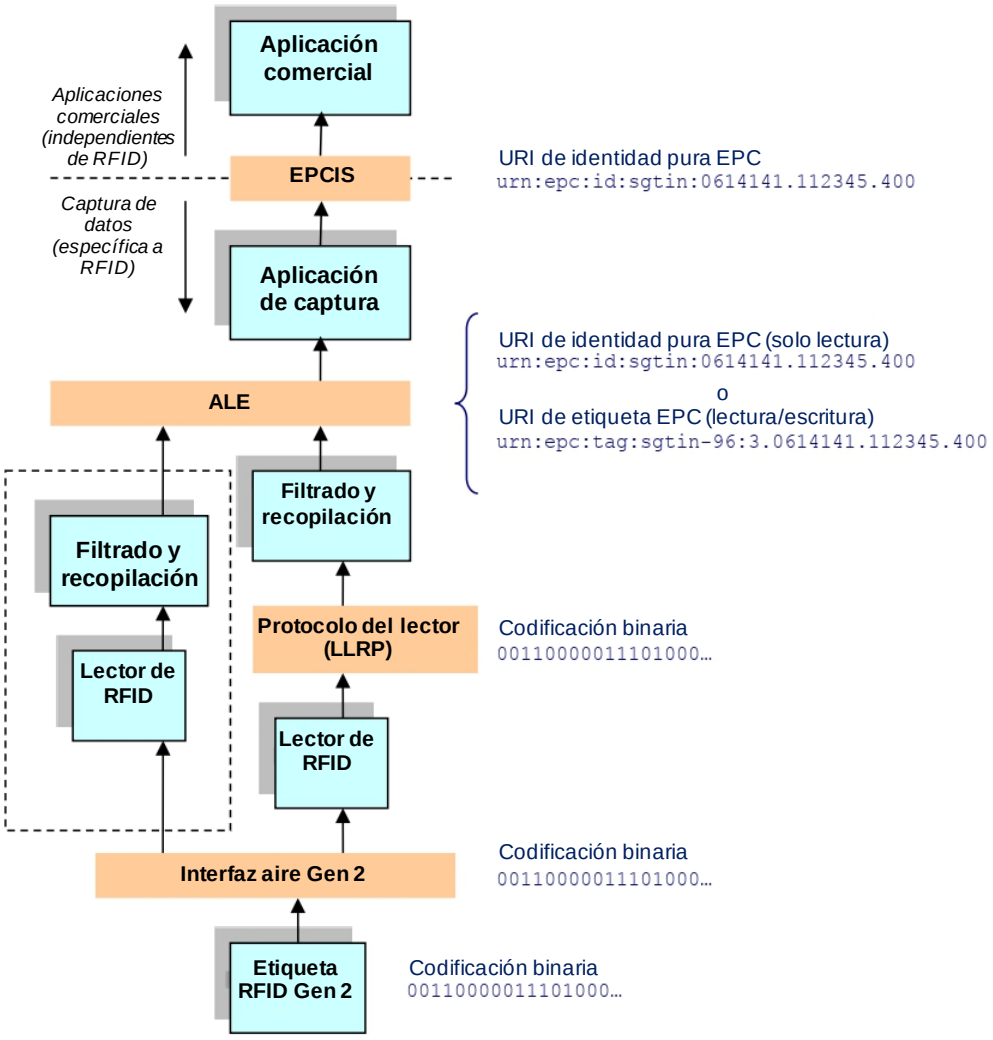
El EPC y las estructuras relacionadas especificadas en este documento están destinadas a utilizarse en diferentes niveles dentro del Marco de la Arquitectura de EPCglobal. Específicamente:

- **URI de identidad pura EPC**: La representación principal de un Código de Producto Electrónico es un identificador de recursos uniforme (URI) de Internet llamado URI de identidad pura EPC. El URI de identidad pura EPC es la forma preferida de indicar un objeto físico específico dentro de las aplicaciones comerciales. El URI de identidad pura también se puede usar en el nivel de captura de datos cuando el EPC debe leerse de una etiqueta RFID u otro portador de datos, en una situación en la que no se necesita que haya información de "control" adicional en una etiqueta RFID.
- **URI de etiqueta EPC**: El banco de memoria EPC de una etiqueta RFID Gen 2 contiene el EPC más "información de control" adicional que se utiliza para guiar el proceso de captura de datos de las etiquetas RFID. El URI de etiqueta EPC es una cadena URI que denota un EPC específico junto con configuraciones específicas para la información de control que se encuentra en el banco de memoria EPC. En otras palabras, el URI de etiqueta EPC es un texto equivalente a todo el contenido del banco de memoria EPC. El URI de etiqueta EPC se utiliza normalmente en el nivel de captura de datos cuando se lee desde una etiqueta RFID en una situación en la que la información de control es de interés para la aplicación de captura. También se utiliza al escribir el banco de memoria EPC de una etiqueta RFID, para especificar completamente el contenido que se va a escribir.
- **Codificación binaria**: El banco de memoria EPC de una etiqueta RFID Gen 2 en realidad contiene una codificación comprimida del EPC e "información de control" adicional en una forma binaria compacta. Existe una traducción de 1 a 1 entre los URI de las etiquetas de EPC y el contenido binario de una etiqueta RFID Gen 2. Por lo general, la codificación binaria solo se encuentra en un nivel muy bajo de software o hardware y se traduce a la forma URI de etiqueta EPC o URI de identidad pura EPC antes de presentarse a la lógica de aplicación.

Tenga en cuenta que el URI de identidad pura EPC es independiente de RFID, mientras que el URI de etiqueta EPC y la Codificación Binaria son específicos de las etiquetas RFID Gen 2 porque incluyen "información de control" específica de RFID además del identificador EPC único.

En la siguiente figura, se ilustra dónde se encuentran normalmente estas estructuras en relación con las capas del Marco de la Arquitectura de EPCglobal.

Figura 4-5 Marco de la Arquitectura de EPCglobal y estructuras de EPC utilizadas en cada nivel



5 Elementos gramaticales comunes

La sintaxis de varias formas de URI definidas en este documento se especifica mediante sintaxis de BNF. A lo largo de esta especificación se emplean los siguientes elementos gramaticales.

```
NumericComponent ::= ZeroComponent | NonZeroComponent
ZeroComponent ::= "0"
NonZeroComponent ::= NonZeroDigit Digit*
PaddedNumericComponent ::= Digit+
PaddedNumericComponentOrEmpty ::= Digit*
Digit ::= "0" | NonZeroDigit
NonZeroDigit ::= "1" | "2" | "3" | "4"
        | "5" | "6" | "7" | "8" | "9"
UpperAlpha ::= "A" | "B" | "C" | "D" | "E" | "F" | "G"
       | "H" | "I" | "J" | "K" | "L" | "M" | "N"
       | "O" | "P" | "Q" | "R" | "S" | "T" | "U"
       | "V" | "W" | "X" | "Y" | "Z"
```

```
LowerAlpha ::= "a" | "b" | "c" | "d" | "e" | "f" | "g"
       | "h" | "i" | "j" | "k" | "l" | "m" | "n"
       | "o" | "p" | "q" | "r" | "s" | "t" | "u"
       | "v" | "w" | "x" | "y" | "z"
OtherChar ::= "!" | "'" | "(" | ")" | "*" | "+" | "," | "-"
      | "." | ":" | ";" | "=" | "_"
UpperHexChar ::= Digit | "A" | "B" | "C" | "D" | "E" | "F"
HexComponent ::= UpperHexChar+
HexComponentOrEmpty ::= UpperHexChar*
Escape ::= "%" HexChar HexChar
HexChar ::= UpperHexChar | "a" | "b" | "c" | "d" | "e" | "f"
GS3A3Char ::= Digit | UpperAlpha | LowerAlpha | OtherChar
      | Escape
GS3A3Component ::= GS3A3Char+
CPRefChar ::= Digit | UpperAlpha | "-" | "%2F" | "%23"
CPRefComponent ::= CPRefChar+
```

El constructo sintáctico `GS3A3Component` se utiliza para representar campos de códigos GS1 que permiten caracteres alfanuméricos y otros caracteres especificados en la Figura 7.12-1 de las Especificaciones Generales de GS1 (consulte el Apéndice A). Debido a restricciones en la sintaxis del URN, según se define en [RFC2141], no todos los caracteres permitidos en las Especificaciones generales de GS1 pueden representarse directamente en un URN. De manera específica, se permiten los caracteres " (comillas dobles), % (porcentaje), & (ampersand), / (barra diagonal), < (menor que), > (mayor que) y ? (signo de interrogación) en las Especificaciones Generales de GS1, pero no pueden incluirse directamente en un URN. Para representar uno de estos caracteres en un URN, se debe utilizar la notación de escape en la que el carácter se representa con un signo de porcentaje, seguido de dos dígitos hexadecimales que proporcionan el código de carácter ASCII para el carácter.

La construcción sintáctica `CPRefComponent` se utiliza para representar campos que permiten mayúsculas alfanuméricas y los caracteres guion, barra diagonal y signo de número/gato. Debido a las restricciones en la sintaxis del URN, según se define en [RFC2141], no todos estos caracteres pueden representarse directamente en un URN. De manera específica, los caracteres # (signo de gato/número) y / (barra diagonal) no pueden incluirse directamente en un URN. Para representar uno de estos caracteres en un URN, se debe utilizar la notación de escape en la que el carácter se representa con un signo de porcentaje, seguido de dos dígitos hexadecimales que proporcionan el código de carácter ASCII para el carácter.

6 URI de EPC

Esta sección especifica la forma de "URI de identidad pura" del EPC, o simplemente el "URI de EPC". El URI de EPC es la forma preferida dentro de un sistema de información para denotar un objeto físico específico.

El URI de EPC es una cadena que tiene la siguiente forma:

```
urn:epc:id:scheme:component1.component2. …
```

donde *scheme* nombra un esquema de EPC y *component1*, *component2* y las siguientes partes son el resto del EPC cuya forma precisa depende del esquema de EPC que se utilice. Los esquemas de EPC disponibles se especifican más abajo en la *Tabla 6-1*, la Sección *6.3*.

El siguiente es un ejemplo de un URI de EPC específico, donde el esquema es sgtin:

```
urn:epc:id:sgtin:0614141.112345.400
```

Cada esquema de EPC proporciona un espacio de nombres de identificadores que se pueden usar para identificar objetos físicos de un tipo particular. En conjunto, los URI de EPC de todos los esquemas son identificadores únicos para cualquier tipo de objeto físico.

6.1 Uso del URI de EPC

El URI de EPC es la forma preferida dentro de un sistema de información para denotar un objeto físico específico.

La estructura del URI de EPC garantiza el carácter único en el mundo del EPC para todo tipo de objetos físicos y aplicaciones.

A fin de preservar esta condición única en el mundo, cada URI de EPC debe usarse completo cuando se solicita un identificador único y no debe dividirse en las partes que lo conforman ni el prefijo `urn:epc:id:` debe abreviarse o eliminarse.

Cuando se hace la pregunta "¿estas dos estructuras de datos se refieren al mismo objeto físico?", donde cada estructura de datos usa un URI de EPC para referirse a un objeto físico, la pregunta puede responderse simplemente comparando las cadenas de URI de EPC completas como se especifica en [RFC3986], Sección 6.2. En la mayoría de los casos, el método de "simple comparación de cadenas" es suficiente, aunque si un URI contiene tripletes de codificación porcentual, es posible que se requiera normalizar los dígitos hexadecimales en mayúsculas y minúsculas según se describe en [RFC3986], Sección 6.2.2.1. La construcción del URI de EPC garantiza el carácter único en todas las categorías de objetos, siempre que el URI se utilice completo.

En otras situaciones, es posible que en aplicaciones busquen explotar la estructura interna de un URI de EPC con fines de filtrado, selección o distribución. Por ejemplo, es posible que una aplicación busque consultar una base de datos para encontrar todos los registros pertenecientes a instancias de un producto específico identificado con un GTIN. Esto equivale a consultar todos los EPC cuyos componentes de Prefijo GS1 de empresa y de referencia del artículo coincidan con un valor determinado, sin tener en cuenta el componente de número de serie. Otro ejemplo se encuentra en el Servicio de Nombre de Objeto (ONS) [ONS1.0.1], que utiliza el primer componente de un EPC para delegar una consulta a un "ONS local" operado por una empresa individual. Esto permite que el sistema de ONS escale de una manera que sería bastante difícil si todos los registros de ONS estuvieran almacenados en una base de datos plana mantenida por una sola organización.

Si bien la estructura interna del EPC puede explotarse para el filtrado, la selección y la distribución, como se ilustra antes, es esencial que el URI de EPC se emplee completo cuando se use como un identificador único.

6.2 Asignación de EPC a objetos físicos

El acto de asignar un nuevo EPC y asociarlo a un objeto físico específico se llama "implementación". Es responsabilidad de las aplicaciones y los procesos comerciales que implementan EPC garantizar que nunca se asigne el mismo EPC a dos objetos físicos diferentes; es decir, garantizar que los EPC implementados sean únicos. Normalmente, las aplicaciones de implementación utilizan bases de datos que registran los EPC que ya se han implementado y los que todavía están disponibles. Por ejemplo, en una aplicación que implementa SGTIN mediante la asignación secuencial de números de serie, dicha base de datos podría registrar el último número de serie utilizado para cada GTIN base.

Debido a que los datos de visibilidad y otros datos comerciales que hacen referencia a unos EPC pueden continuar existiendo mucho después de que el objeto físico deje de existir, lo ideal es que nunca se reutilice un EPC para referirse a un objeto físico diferente, incluso si ocurre después de que el objeto original deje de existir. Sin embargo, hay ciertas situaciones en las que esto no es posible, algunas de las cuales se indican más abajo. Por ello, las aplicaciones que procesan datos históricos empleando EPC deberían estar preparadas para la posibilidad de que un EPC pueda reutilizarse con el tiempo para referirse a objetos físicos diferentes, a menos que se sepa que la aplicación funciona en un entorno donde se evita dicha reutilización.

Siete de los esquemas de EPC especificados en este documento corresponden a llaves GS1, por lo que los EPC de esos esquemas se utilizan para identificar objetos físicos con una llave GS1 correspondiente. Al asignar estos tipos de EPC a objetos físicos, se deben seguir todas las reglas GS1 pertinentes además de las reglas especificadas en este documento. Esto incluye las Especificaciones Generales de GS1 [GS1GS], el Estándar de Gestión de GTIN, etc. En particular, un EPC de este tipo solo puede ser implementado por el licenciatario de un Prefijo GS1 de empresa que es parte del EPC o si el licenciatario del Prefijo GS1 de empresa le ha delegado la autoridad para hacerlo.

6.3 Sintaxis del URI de EPC

Esta sección especifica la sintaxis de un URI de EPC.

La sintaxis formal para el URI de EPC es la siguiente:

```
EPC-URI ::= SGTIN-URI | SSCC-URI | SGLN-URI | GRAI-URI | GIAI-URI |
      GSRN-URI | GDTI-URI | CPI-URI | SGCN-URI | GINC-URI | GS1N-URI
      ITIP-URI | UPUI-URI | PGLN-URI | GID-URI | DOD-URI | ADI-URI |
      BIC-URI
```

donde las diversas alternativas en el lado derecho se especifican en las secciones que siguen.

Cada esquema de URI de EPC se especifica en una de las siguientes subsecciones, de la siguiente manera:

Figura 6-1 Esquemas de EPC y donde se define la forma de identidad pura.

Esquema de EPC	Especificado en	Llave GS1 correspondiente	Uso típico
sgtin	Sección 6.3.1	GTIN (con número de serie agregado)	Artículo comercial
sscc	Sección 6.3.2	SSCC	Unidad de logística
sgln	Sección 6.3.3	GLN (con o sin extensión adicional)	Localización[2]
grai	Sección 6.3.4	GRAI (número de serie obligatorio)	Activo retornable
giai	Sección 6.3.5	GIAI	Activo fijo
gsrn	Sección 6.3.6	GSRN - Receptor	Hospitalización o membresía en un club.
gsrnp	Sección 6.3.7	GSRN - Proveedor	Cuidador médico o club de lealtad
gdti	Sección 6.3.8	GDTI (número de serie obligatorio)	Documento
cpi	Sección 6.3.9	[ninguno]	Industrias técnicas (p. ej., el sector automotriz) para la identificación única de partes y componentes
sgcn	Sección 6.3.10	GCN (número de serie obligatorio)	Cupón
ginc	Sección 6.3.11	GINC	Agrupación lógica de mercancías destinadas a transportarse en conjunto, asignada por un transportista
GS1n	Sección 6.3.12	GS1N	Agrupación lógica de unidades logísticas que viajan bajo aviso de envío y/o acuse de recibo
itip	Sección 6.3.13	AI (8006) combinado con AI (21)	Una de las múltiples partes que abarcan y están subordinadas a un todo (que, a su vez, se identifica mediante un SGTIN o la combinación de AI 01 + 21).
upui	Sección 6.3.14	GTIN y TPX	Identificación del paquete para combatir el comercio ilícito
pgln	Sección 6.3.15	GLN de la parte - AI (417)	Identificación del operador económico; identificación de la parte propietaria o poseedora en la Cadena de Custodia (CoC)/Cadena de Propiedad (CoO).
gid	Sección 6.3.16	[ninguno]	Sin especificar

[2] Si bien los GLN se pueden usar para identificar tanto ubicaciones como partes, el SGLN corresponde solo a AI 414, que [GS1GS] especifica que se usará para identificar ubicaciones y no partes.

Esquema de EPC	Especificado en	Llave GS1 correspondiente	Uso típico
usdod	Sección 6.3.17	[ninguno]	Cadena de suministro del Departamento de Defensa de los EE. UU.
adi	Sección 6.3.18	[ninguno]	Sector aeroespacial y de defensa para la identificación única de aeronaves y otras partes y artículos
bic	Sección 6.3.19	[ninguno]	Contenedores de transporte intermodal

6.3.1 Número Global de Artículo Comercial Serializado (SGTIN)

El esquema EPC Número Global de Artículo Comercial Serializado (SGTIN) se utiliza para asignar una identidad única a una instancia de un artículo comercial, como una instancia específica de un producto o SKU.

Sintaxis general:

`urn:epc:id:sgtin:`*`CompanyPrefix.ItemRefAndIndicator.SerialNumber`*

Ejemplo:

`urn:epc:id:sgtin:0614141.112345.400`

Sintaxis:

```
SGTIN-URI ::= "urn:epc:id:sgtin:" SGTINURIBody

SGTINURIBody ::= 2*(PaddedNumericComponent ".") GS3A3Component
```

La cantidad de caracteres en los dos campos `PaddedNumericComponent` debe sumar 13 (sin incluir ninguno de los caracteres de punto).

El campo de Número de Serie del SGTIN-URI se expresa como `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie del Identificador de Aplicación 21 de acuerdo con las Especificaciones Generales de GS1;[3] sin embargo, los SGTIN-URI que se derivan de las codificaciones de etiquetas de 96 bits tendrán Números de Serie que consistan únicamente de dígitos y que no tengan ceros a la izquierda (a menos que el número de serie completo consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección 12.3.1.

El SGTIN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora o sus delegados. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GTIN GS1. Consulte la Sección 7.3.2 para ver el caso de un GTIN-8.
- La **Referencia del Artículo**, asignada por la entidad gestora a una clase de objeto en particular. La Referencia del Artículo tal como aparece en el URI de EPC se obtiene del GTIN, concatenando el dígito indicador del GTIN (o un carácter de relleno cero, si el URI de EPC se deriva de un GTIN-8, GTIN-12 o GTIN-13) con los dígitos de Referencia del Artículo y tratando el resultado como una única cadena numérica. Consulte la Sección 7.3.2 para ver el caso de un GTIN-8.
- El **Número de Serie**, asignado por la entidad gestora a un objeto individual. El número de serie no es parte del GTIN, pero formalmente sí lo es del SGTIN.

6.3.2 Código Seriado de Contenedor de Envío (SSCC)

El esquema de EPC Código Seriado de Contenedor de Envío (SSCC) se utiliza para asignar una identidad única a una unidad de manipulación de logística, como el contenido agregado de un contenedor de envío o una carga de tarima.

---

[3] Como se especifica en la Sección 7.1, el número de serie en el SGTIN se define actualmente como equivalente a AI 21 en las Especificaciones Generales de GS1. Esta equivalencia se encuentra actualmente en debate dentro de GS1 y es posible que se revise en versiones futuras del Estándar de Datos de Etiquetas EPC.

Sintaxis general:
urn:epc:id:sscc:*CompanyPrefix.SerialReference*

Ejemplo:

urn:epc:id:sscc:0614141.1234567890

Gramática:

SSCC-URI ::= "urn:epc:id:sscc:" SSCCURIBody

SSCCURIBody ::= PaddedNumericComponent "." PaddedNumericComponent

La cantidad de caracteres en los dos campos PaddedNumericComponent debe sumar 17 (sin incluir ninguno de los caracteres de punto).

El SSCC consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave SSCC GS1.
- La **Referencia de la Serie**, asignada por la entidad gestora a una unidad de manipulación de logística en particular. La Referencia de la Serie tal como aparece en el URI de EPC se obtiene del SSCC, concatenando el dígito de extensión del SSCC con los dígitos de Referencia de la Serie y tratando el resultado como una sola cadena numérica.

6.3.3 Número Global de Localización con o sin Extensión (SGLN)

El esquema de EPC SGLN se utiliza para asignar una identidad única a una localización física, como un edificio específico o una unidad específica de estanterías dentro de un almacén.

Sintaxis general:

urn:epc:id:sgln:*CompanyPrefix.LocationReference.Extension*

Ejemplo:

urn:epc:id:sgln:0614141.12345.400

Sintaxis:

SGLN-URI ::= "urn:epc:id:sgln:" SGLNURIBody

SGLNURIBody ::= PaddedNumericComponent "." PaddedNumericComponentOrEmpty "." GS3A3Component

La cantidad de caracteres en los dos campos PaddedNumericComponent debe sumar 12 (sin incluir ninguno de los caracteres de punto).

El campo de Extensión del SGLN-URI se expresa como un GS3A3Component, que permite la representación de todos los caracteres permitidos en la Extensión del Identificador de Aplicación 254 de acuerdo con las Especificaciones Generales de GS1. Sin embargo, los SGLN-URI que se derivan de codificaciones de etiquetas de 96 bits tendrán extensiones que consistan únicamente de dígitos y que no tengan ceros a la izquierda (a menos que toda la extensión consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El SGLN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GLN GS1.
- La **Referencia de la Localización**, asignada de forma única por la entidad gestora a una localización física específica.
- La **Extensión GLN**, asignada por la entidad gestora a una localización única individual. Si la extensión GLN completa es solo un dígito cero, indica que SGLN representa un GLN sin extensión.

Reglas no normativas: Explicación (no normativa): Tenga en cuenta que la letra "S" en el término "SGLN" no significa "serializado" como lo hace en SGTIN. Esto se debe a que un GLN sin una extensión también identifica una localización única, a diferencia de una clase de localizaciones, por lo que tanto el GLN como el GLN con extensión pueden considerarse identificadores "serializados". El término SGLN simplemente distingue la forma de EPC, que puede usarse para un GLN por sí mismo o un GLN con extensión, a partir del término GLN que siempre se refiere al identificador GLN no extendido. La letra "S" no representa nada.

6.3.4 Identificador Global de Activos Retornables (GRAI)

El esquema de EPC Identificador Global de Activos Retornables (GRAI) se utiliza para asignar una identidad única a un activo retornable específico, como un contenedor de envío reutilizable o una tarima.

Sintaxis general:

`urn:epc:id:grai: CompanyPrefix. AssetType. SerialNumber`

Ejemplo:

`urn:epc:id:grai:0614141.12345.400`

Sintaxis:

```
GRAI-URI ::= "urn:epc:id:grai:" GRAIURIBody

GRAIURIBody ::= PaddedNumericComponent "." PaddedNumericComponentOrEmpty "."
GS3A3Component
```

La cantidad de caracteres en los dos campos PaddedNumericComponent debe sumar 12 (sin incluir ninguno de los caracteres de punto).

El campo de Número de Serie del GRAI-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie de acuerdo con las Especificaciones Generales de GS1. Sin embargo, los GRAI-URI que se derivan de codificaciones de etiquetas de 96 bits tendrán números de serie que consistan únicamente de dígitos y que no tengan ceros a la izquierda (a menos que el número de serie completo consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El GRAI consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GRAI GS1.
- El **Tipo de Activo**, asignado por la entidad gestora a una clase particular de activo.
- El **Número de Serie**, asignado por la entidad gestora a un objeto individual. Debido a que un EPC siempre se refiere a un objeto físico específico en lugar de una clase de activo, el número de serie es obligatorio en el EPC GRAI.

6.3.5 Identificador Global Individual de Activo (GIAI)

El esquema de EPC Identificador Global Individual de Activo (GIAI) se utiliza para asignar una identidad única a un activo específico, como un montacargas o una computadora.

Sintaxis general:

`urn:epc:id:giai: CompanyPrefix. IndividulAssetReference`

Ejemplo:

`urn:epc:id:giai:0614141.12345400`

Sintaxis:

```
GIAI-URI ::= "urn:epc:id:giai:" GIAIURIBody
```

```
GIAIURIBody ::= PaddedNumericComponent "." GS3A3Component
```

El campo de Referencia de Activo Individual del GIAI-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie de acuerdo con las Especificaciones Generales de GS1. Sin embargo, los GIAI-URI que se derivan de codificaciones de etiquetas de 96 bits tendrán números de serie que consistan únicamente de dígitos y que no tengan ceros a la izquierda (a menos que el número de serie completo consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El GIAI consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. El Prefijo de empresa es el mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GIAI GS1.
- La **Referencia Individual del Activo**, asignada de forma única por la entidad gestora a un activo específico.

6.3.6 Número Global de Relación de Servicio - Receptor (GSRN)

El esquema de EPC Número Global de Relación de Servicio (GSRN) se utiliza para asignar una identidad única a un receptor de un servicio.

Sintaxis general:

`urn:epc:id:gsrn:`*`CompanyPrefix.ServiceReference`*

Ejemplo:

```
urn:epc:id:gsrn:0614141.1234567890
```

Sintaxis:

```
GSRN-URI ::= "urn:epc:id:gsrn:" GSRNURIBody

GSRNURIBody ::= PaddedNumericComponent "." PaddedNumericComponent
```

La cantidad de caracteres en los dos campos `PaddedNumericComponent` debe sumar 17 (sin incluir ninguno de los caracteres de punto).

El GSRN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GSRN GS1.
- La **Referencia de Servicio**, asignada por la entidad gestora a un receptor de un servicio en particular.

6.3.7 Número Global de Relación de Servicio - Proveedor (GSRNP)

El esquema de EPC Número Global de Relación de Servicio - Proveedor (GSRNP) se utiliza para asignar una identidad única a un proveedor de servicio.

Sintaxis general:

`urn:epc:id:gsrnp:`*`CompanyPrefix. ServiceReference`*

Ejemplo:

```
urn:epc:id:gsrnp:0614141.1234567890
```

Sintaxis:

```
GSRNP-URI ::= "urn:epc:id:gsrnp:" GSRNURIBody

GSRNPURIBody ::= PaddedNumericComponent "." PaddedNumericComponent
```

La cantidad de caracteres en los dos campos PaddedNumericComponent debe sumar 17 (sin incluir ninguno de los caracteres de punto).

El GSRNP consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GSRN GS1.
- La **Referencia de Servicio**, asignada por la entidad gestora a un proveedor de servicio en particular.

6.3.8 Identificador Global de Tipo de Documento (GDTI)

El esquema de EPC Identificador Global de Tipo de Documento (GDTI) se utiliza para asignar una identidad única a un documento específico, como los de registro de la propiedad, una póliza de seguro y otros documentos.

Sintaxis general:

`urn:epc:id:gdti:`*`CompanyPrefix.DocumentType.SerialNumber`*

Ejemplo:

`urn:epc:id:gdti:0614141.12345.400`

Sintaxis:

```
GDTI-URI ::= "urn:epc:id:gdti:" GDTIURIBody

GDTIURIBody ::= PaddedNumericComponent "." PaddedNumericComponentOrEmpty
"."GS3A3Component
```

La cantidad de caracteres en los dos campos `PaddedNumericComponent` debe sumar 12 (sin incluir ninguno de los caracteres de punto).

El campo de Número de Serie del GDTI-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie de acuerdo con las Especificaciones Generales de GS1. Sin embargo, los GDTI-URI que se derivan de codificaciones de etiquetas de 96 bits tendrán números de serie que no tengan ceros a la izquierda (a menos que el número de serie completo consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El GDTI consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GDTI GS1.
- El **Tipo de Documento**, asignado por la entidad gestora a una clase particular de documento.
- El **Número de Serie**, asignado por la entidad gestora a un documento individual. Debido a que un EPC siempre se refiere a un documento específico en lugar de una clase de documento, el número de serie es obligatorio en el EPC GDTI.

6.3.9 Identificador de Componente / Pieza (CPI)

El EPC Identificador de Componente / Pieza (CPI) está diseñado para que lo utilicen las industrias técnicas (incluido el sector automotriz) para la identificación única de partes o componentes.

La construcción del EPC CPI proporciona un mecanismo para codificar directamente identificadores únicos en las etiquetas RFID y utilizar las representaciones URI en otras capas de la arquitectura de EPCglobal.

Sintaxis general:

`urn:epc:id:cpi:`*`CompanyPrefix.ComponentPartReference.Serial`*

Ejemplo:

`urn:epc:id:cpi:0614141.123ABC.123456789`

`urn:epc:id:cpi:0614141.123456.123456789`

Sintaxis:

```
CPI-URI ::= "urn:epc:id:cpi:" CPIURIBody
```

```
CPIURIBody ::= PaddedNumericComponent "." CPRefComponent "." NumericComponent
```

El campo de Referencia de componente/pieza del CPI-URI se expresa como `CPRefComponent`, que permite la representación de todos los caracteres permitidos en la Referencia de componente/pieza de acuerdo con las Especificaciones Generales de GS1. Sin embargo, los CPI-URI que se derivan de codificaciones de etiquetas de 96 bits tendrán Referencias de Componentes/Partes que consistan únicamente en dígitos, sin ceros a la izquierda y cuya longitud sea menor o igual a 15 menos la longitud del Prefijo GS1 de empresa. Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El CPI consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora o sus delegados.
- La **Referencia de componente/pieza**, asignada por la entidad gestora a una clase de objeto en particular.
- El **Número de Serie**, asignado por la entidad gestora a un objeto individual.

La entidad gestora o sus delegados se aseguran de que cada CPI no se emita a más de un componente o parte física. Normalmente esto se logra asignando una referencia de componente/pieza para designar un conjunto de instancias de una parte que comparten la misma forma, ajuste o función y luego emitiendo valores de número de serie de forma única dentro de cada valor de referencia de componente/pieza para distinguir entre tales instancias.

6.3.10 Número de Cupón Global Serializado (SGCN)

El esquema de EPC Número de Cupón Global (GCN) se utiliza para asignar una identidad única a un cupón.

Sintaxis general:

`urn:epc:id:sgcn:`*`CompanyPrefix.CouponReference.SerialComponent`*

Ejemplo:

`urn:epc:id:sgcn:4012345.67890.04711`

Sintaxis:

```
SGCN-URI ::= "urn:epc:id:sgcn:" SGCNURIBody
```

```
SGCNURIBody ::= PaddedNumericComponent "." PaddedNumericComponentOrEmpty "."
PaddedNumericComponent
```

El número de caracteres en el primer campo `PaddedNumericComponent` y el campo `PaddedNumericComponentOrEmpty` debe sumar 12 (sin incluir ninguno de los caracteres de punto).

El campo Componente de Serie del SGCN-URI se expresa como un `PaddedNumericComponent`, que puede contener hasta 12 dígitos, incluidos los ceros a la izquierda, según las Especificaciones Generales de GS1. El SGCN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GCN GS1.
- La **Referencia de Cupón**, asignada por la entidad gestora del cupón.
- El **Componente de Serie**, asignado por la entidad gestora a una instancia única del cupón. Debido a que un EPC siempre se refiere a un cupón específico en lugar de a una clase de cupón, el número de serie es obligatorio en el EPC SGCN.

6.3.11 Número Global de Identificación de Consignatario (GINC)

El esquema de EPC Número Global de Identificación de Consignatario (GINC) se utiliza para asignar una identidad única a una agrupación lógica de mercancías (una o más entidades físicas) que se ha enviado a un cosignatario y que se transportará en conjunto.

Sintaxis general:

`urn:epc:id:ginc:CompanyPrefix.ConsignmentReference`

Ejemplo:

`urn:epc:id:ginc:0614141.xyz3311cba`

Sintaxis:

```
GINC-URI ::= "urn:epc:id:ginc:" GINCURIBody

GINCURIBody ::= PaddedNumericComponent "." GS3A3Component
```

El campo de Referencia de Consignatario del GINC-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie de acuerdo con las Especificaciones Generales de GS1.

El GINC consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. El Prefijo de empresa es el mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GINC GS1.
- La **Referencia de Consignatario**, asignada de forma exclusiva por el transportista.

6.3.12 Número Global de Identificación de Envío (GS1N)

El esquema de EPC Número Global de Identificación de Envío (GSIN) se utiliza para asignar una identidad única a una agrupación lógica de unidades logísticas con el objetivo de un envío de transporte desde ese consignador (vendedor) hasta el consignatario (comprador).

Sintaxis general:

`urn:epc:id:gsin:CompanyPrefix.ShipperReference`

Ejemplo:

`urn:epc:id:GS1n:0614141.123456789`

Sintaxis:

```
GS1N-URI ::= "urn:epc:id:GS1n:" GS1NURIBody

GS1NURIBody ::= PaddedNumericComponent "." PaddedNumericComponent
```

El número de caracteres en los dos campos PaddedNumericComponent debe sumar 17 (sin incluir el carácter de punto).

La familia de GS1N consta de las siguientes variaciones:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GS1N GS1.
- La **Referencia del Remitente**, asignada por el consignador (vendedor) de las mercancías.

6.3.13 Pieza de Artículo Comercial Individual (ITIP)

El esquema de EPC Pieza de Artículo Comercial Individual (ITIP) se utiliza para asignar una identidad única a un elemento subordinado de un artículo comercial (por ejemplo, zapato izquierdo y zapato derecho, pantalón y chaqueta de traje, artículo comercial de "hágalo usted mismo" que consta de varias unidades físicas), el último de los cuales comprende múltiples piezas.

Sintaxis general:

`urn:epc:id:itip:CompanyPrefix.ItemRefAndIndicator.Piece.Total.SerialNumber.`

Ejemplo:

`urn:epc:id:itip:4012345.012345.01.02.987`

Sintaxis:

```
ITIP-URI ::= "urn:epc:id:itip:" ITIPURIBody

ITIPURIBody ::= 4*(PaddedNumericComponent ".") GS3A3Component
```

El número de caracteres en los primeros dos campos `PaddedNumericComponent` debe sumar 13 (sin incluir ninguno de los caracteres de punto).

El número de caracteres en cada uno de los dos últimos campos `PaddedNumericComponent` debe ser exactamente 2 (sin incluir ninguno de los caracteres de punto).

El número combinado de caracteres en los cuatro campos `PaddedNumericComponent` debe sumar 17 (sin incluir ninguno de los caracteres de punto).

El campo de Número de Serie del ITIP-URI se expresa como `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de Serie del Identificador de Aplicación 21 de acuerdo con las Especificaciones Generales de GS1;[4] sin embargo, los ITIP-URI que se derivan de las codificaciones de etiquetas de 110 bits tendrán Números de Serie que consistan únicamente de dígitos y que no tengan ceros a la izquierda (a menos que el número de serie completo consista en un solo dígito cero). Estas limitaciones se describen en los procedimientos de codificación y en la Sección *12.3.1*.

El ITIP consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora o sus delegados. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GTIN GS1. Consulte la Sección *7.3.2* para ver el caso de un GTIN-8.
- La **Referencia del Artículo**, asignada por la entidad gestora a una clase de objeto en particular. La Referencia del Artículo tal como aparece en el URI de EPC se obtiene del GTIN, concatenando el dígito indicador del GTIN (o un carácter de relleno cero, si el URI de EPC se deriva de un GTIN-8, GTIN-12 o GTIN-13) con los dígitos de Referencia del Artículo y tratando el resultado como una única cadena numérica. Consulte la Sección *7.3.2* para ver el caso de un GTIN-8.
- El número de **Pieza**
- La Cantidad **Total** de Piezas subordinada al GTIN
- El **Número de Serie**, asignado por la entidad gestora a un objeto individual. El número de serie no es parte del GTIN, pero formalmente sí lo es tanto del SGTIN como del ITIP.

6.3.14 Identificador a Nivel de las Unidades de Envasado (UPUI)

El esquema de EPC Identificador a Nivel de las Unidades de Envasado se utiliza para identificar de forma única un artículo individual para la trazabilidad del tabaco de acuerdo con UE 2018/574.

Sintaxis general:

`urn:epc:id:upui:`*`CompanyPrefix.ItemRefAndIndicator.TPX`*

Ejemplo:

```
urn:epc:id:upui:1234567.089456.51qIgY)%3C%26Jp3*j7'SDB
```

Sintaxis:

```
UPUI-URI ::= "urn:epc:id:upui:" UPUI-URIBody

UPUI-URIBody ::= 2*(PaddedNumericComponent ".") GS3A3Component
```

El número de caracteres en los primeros dos campos `PaddedNumericComponent` debe sumar 13 (sin incluir ninguno de los caracteres de punto).

El campo *TPX* del UPUI-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Identificador de Aplicación (235), Extensión Serializada del GTIN Controlada por Terceros, de acuerdo con las Especificaciones Generales de GS1.[5]

El UPUI consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora o sus delegados. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GTIN GS1. Consulte la Sección 7.3.2 para ver el caso de un GTIN-8.
- La **Referencia del Artículo**, asignada por la entidad gestora a una clase de objeto en particular. La Referencia del Artículo tal como aparece en el URI de EPC se obtiene del GTIN, concatenando el dígito indicador del GTIN (o un carácter de relleno cero, si el URI de EPC se deriva de un GTIN-8, GTIN-12 o GTIN-13) con los dígitos de Referencia del Artículo y tratando el resultado como una única cadena numérica. Consulte la Sección 7.3.2 para ver el caso de un GTIN-8.
- La **extensión serializada de GTIN controlada por terceros** , asignada por una entidad administradora de terceros a un objeto individual para identificar de manera única un artículo individual para la trazabilidad del tabaco de acuerdo con la EU 2018/574.

6.3.15 Número Global de Localización de la Parte (PGLN)

El esquema de EPC PGLN se utiliza para asignar una identidad única a una parte, como un operador económico o un centro de costos.

Sintaxis general:

`urn:epc:id:pgln:`*`CompanyPrefix.PartyReference`*

Ejemplo:

`urn:epc:id:pgln:1234567.89012`

Sintaxis:

```
PGLN-URI ::= "urn:epc:id:pgln:" PGLNURIBody

PGLNURIBody ::= PaddedNumericComponent "." PaddedNumericComponentOrEmpty
```

La cantidad de caracteres en los dos campos PaddedNumericComponent debe sumar 12 (sin incluir ninguno de los caracteres de punto).

El PGLN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GLN GS1.
- La **referencia de parte**, asignada de forma única por la entidad gestora a una parte específica.

6.3.16 Identificador general (GID)

El esquema de EPC Identificador General (GID) es independiente de cualquier especificación o esquema de identidad fuera del Estándar de Datos de Etiquetas de EPCglobal.

Sintaxis general:

`urn:epc:id:gid:`*`ManagerNumber.ObjectClass.SerialNumber`*

Ejemplo:

`urn:epc:id:gid:95100000.12345.400`

Sintaxis:

```
GID-URI ::= "urn:epc:id:gid:" GIDURIBody

GIDURIBody ::= 2*(NumericComponent ".") NumericComponent
```

El GID consta de los siguientes elementos:

- El **Número de Gerente General** identifica una entidad organizativa (esencialmente una empresa, un gerente u otra organización) que es responsable de mantener los números en los campos siguientes: Clase de Objeto y Número de Serie. GS1 asigna el Número de Gerente General a una entidad y se asegura de que cada uno sea único. Tenga en cuenta que un Número de Gerente General ***no*** es un Prefijo GS1 de empresa. El Número de Gerente General solo se puede usar en los EPC GID.
- La **Clase de Objeto** es utilizada por una entidad de gestión de EPC para identificar una clase o "tipo" de cosa. Estos números de clase de objeto, por supuesto, deben ser únicos dentro de cada dominio de Número de Gerente General.
- Por último, el código del **Número de Serie** o el número de serie es único dentro de cada clase de objeto. En otras palabras, la entidad gestora es responsable de asignar números de serie únicos y sin repetir para cada instancia dentro de cada clase de objeto.

6.3.17 Identificador del Departamento de Defensa de los EE. UU. (DOD)

El identificador del Departamento de Defensa de los EE. UU. es definido por el Departamento de Defensa de los Estados Unidos. Esta construcción de datos de etiqueta se puede utilizar para codificar etiquetas de Clase 1 de 96 bits para el envío de mercancías al Departamento de Defensa de los Estados Unidos por parte de un proveedor al que ya se le ha asignado un código CAGE (Entidad Comercial y Gubernamental).

En el momento de la redacción de este texto, los detalles de la información que se codifica en estos campos se explican en un documento titulado "United States Department of Defense Supplier's Passive RFID Information Guide", que se puede obtener en el sitio web del Departamento de Defensa de los Estados Unidos (http://www.dodrfid.org/supplierguide.htm).

Tenga en cuenta que la Guía del DOD reconoce explícitamente el valor de los estándares vigentes entre sucursales a nivel mundial y advierte que "los proveedores que son suscriptores de EPCglobal y poseen un Prefijo de empresa [de GS1] único pueden usar cualquiera de los tipos de identidad y las instrucciones de codificación descritas en el documento de Estándar de Datos de Etiquetas EPC™ para codificar etiquetas".

Sintaxis general:

urn:epc:id:usdod:*CAGEOrDODAAC.SerialNumber*

Ejemplo:

urn:epc:id:usdod:2S194.12345678901

Sintaxis:

```
DOD-URI ::= "urn:epc:id:usdod:" DODURIBody

DODURIBody ::= CAGECodeOrDODAAC "." DoDSerialNumber

CAGECodeOrDODAAC ::= CAGECode | DODAAC

CAGECode ::= CAGECodeOrDODAACChar*5

DODAAC ::= CAGECodeOrDODAACChar*6

DoDSerialNumber ::= NumericComponent

CAGECodeOrDODAACChar ::= Digit | "A" | "B" | "C" | "D" | "E" | "F" | "G" |
"H" | "J" | "K" | "L" | "M" | "N" | "P" | "Q" | "R" | "S" | "T" | "U" | "V"
| "W" | "X" | "Y" | "Z"
```

6.3.18 Identificador Aeroespacial y de Defensa (ADI)

El EPC Identificador Aeroespacial y de Defensa (ADI) de longitud variable está diseñado para ser utilizado por el sector aeroespacial y de defensa para la identificación única de partes o artículos. Las construcciones de identificadores únicos existentes se definen en el estándar Spec 2000 [SPEC2000] de la Asociación de Transporte Aéreo (ATA) y en la Guía para la Identificación Única de Artículos [UID] del Departamento de Defensa de los EE. UU. La construcción del EPC ADI proporciona un mecanismo para codificar directamente identificadores únicos en las etiquetas RFID y utilizar las representaciones URI en otras capas de la arquitectura de EPCglobal.

Dentro de las construcciones de identificación del sector aeroespacial y de defensa respaldadas por el EPC ADI, las empresas se identifican de forma única por su código de Entidad Comercial y Gubernamental (CAGE) o por su Código de Dirección de Actividades del Departamento de Defensa (DODAAC). El código NATO CAGE (NCAGE) es emitido por la OTAN/Comité Aliado 135 y es estructuralmente equivalente a un código CAGE (cinco caracteres alfanuméricos en mayúsculas excluyendo las letras mayúsculas I y O) y no entra en conflicto con los códigos CAGE emitidos por el Servicio de Información de Logística de Defensa de los EE. UU. (DLIS). Tenga en cuenta que en el resto de esta sección todas las referencias a CAGE se aplican igualmente a NCAGE.

El Spec 2000 de la ATA define que se puede construir un identificador único mediante la combinación del código CAGE o DODAAC junto con:

- Un número de serie (SER) que se asigna de forma única dentro del código CAGE o DODAAC o
- Un número de parte original (PNO) que es único dentro del código CAGE o DODAAC y un número de serie secuencial (SEQ) que se asigna de forma exclusiva dentro de ese número de parte original.

La Guía para la Identificación Única de Artículos del DOD de los EE. UU. define una serie de métodos aceptables para construir identificadores de artículos únicos (UII). Los UII que se pueden representar mediante el EPC Identificador Aeroespacial y de Defensa son los que se construyen mediante la combinación de un código CAGE o DODAAC junto con:

- un número de serie que es único dentro del identificador empresarial. (construcción de UII #1)
- un número de parte original y un número de serie que es único dentro del número de parte original (un subconjunto de constructo de UII #2)

Tenga en cuenta que los lineamientos para la UID del DOD de los EE. UU. reconocen una serie de identificadores únicos basados en llaves de identificador GS1 como UID válidas. En particular, el SGTIN (GTIN + número de serie), el GIAI y el GRAI con serialización completa se reconocen como UID válidos. Estos pueden representarse en forma de EPC utilizando los esquemas de EPC SGTIN, GIAI y GRAI como se especifica en las Secciones 6.3.1, 6.3.5 y 6.3.4, respectivamente; el esquema de EPC ADI ***no*** se utiliza con este fin. Por el contrario, los lineamientos para la UID del DOD de los EE. UU. también reconocen una amplia gama de identificadores empresariales emitidos por diversas agencias emisoras distintas de las descritas anteriormente; dichas UID no tienen una representación de EPC correspondiente.

Para identificar con RFID las partes de aeronave que tradicionalmente no se serializan o no es necesario serializarlas para otros fines, se puede utilizar el esquema de EPC ADI para asignar un identificador único a una parte. En esta situación, el primer carácter del componente del número de serie de EPC ADI DEBE ser un solo carácter '#'. Se utiliza así para indicar que el número de serie no corresponde con el número de serie de una parte tradicionalmente serializada, porque el carácter '#' no puede aparecer dentro de los valores asociados a los identificadores de elementos de texto SER o SEQ en el estándar Spec 2000 de la ATA.

Para las parten que tradicionalmente se serializan/requieren serializarse para fines distintos de tener un identificador RFID único, y para cualquier uso que figure dentro de los lineamientos para la UID del DOD de los EE. UU., el carácter '#' NO DEBE aparecer dentro del elemento de número de serie.

El estándar Spec 2000 de la ATA recomienda que las empresas serialicen exclusivamente dentro de su código CAGE. Para las empresas que se serializan exclusivamente dentro de su código CAGE o DODAAC, se DEBE utilizar una cadena de longitud cero en lugar del elemento de Número de Parte Original al construir un EPC.

Sintaxis general:

`urn:epc:id:adi: CAGEOrDODAAC. OriginalPartNumber. Seriado`

Ejemplos:

`urn:epc:id:adi:2S194..12345678901`

`urn:epc:id:adi:W81X9C.3KL984PX1.2WMA52`

Sintaxis:

```
ADI-URI ::= "urn:epc:id:adi:" ADIURIBody

ADIURIBody ::= CAGECodeOrDODAAC "." ADIComponent "." ADIExtendedComponent ADIComponent
::= ADIChar*

ADIExtendedComponent ::= "%23"? ADIChar+

ADIChar ::= UpperAlpha | Digit | OtherADIChar

OtherADIChar ::= "-" | "%2F"

CAGECodeOrDODAAC se define en la Sección 6.3.14.
```

6.3.19 Código de Contenedor BIC (BIC)

(fuente: https://en.wikipedia.org/wiki/ISO 6346#IdentificationSystem)

ISO 6346 es una ***norma internacional*** que cubre la codificación, la identificación y el marcado de ***contenedores (de envío) intermodales*** utilizados en ***el transporte de carga intermodal en contenedores.*** La norma establece un sistema de identificación visual para cada contenedor que incluya un número de serie único (con un ***dígito de verificación***), el propietario, un código de país, un tamaño, el tipo y la categoría de equipo, así como cualquier marca operativa. El estándar está gestionado por la ***Oficina Internacional de Contenedores y Transporte Intermodal*** (BIC).

El BIC consta de los siguientes elementos:

- El **código de propietario** consta de tres letras mayúsculas del alfabeto latino para indicar el propietario u operador principal del contenedor. Dicho código debe registrarse en el ***Bureau International des Conteneurs*** en París para garantizar su carácter único en todo el mundo.
- El **identificador de categoría de equipo** consta de una de las siguientes letras mayúsculas del alfabeto latino:
  - U para todos los contenedores de carga
  - J para los equipos relacionados con contenedores de carga desmontables
  - Z para remolques y chasis
- El **número de serie** consta de 6 dígitos numéricos, asignados por el propietario u operador, que identifican exclusivamente el contenedor dentro de la flota de ese propietario/operador.
- El **digito de verificación** consta de un dígito numérico que proporciona un medio para validar las precisiones de grabación y transmisión del código y número de serie del propietario.

Los elementos individuales del BIC no están separados por puntos (".") en la sintaxis del URI de EPC.

Sintaxis general:

urn:epc:id:bic:*BICcontainerCode*

Ejemplo:

urn:epc:id:bic:CSQU3054383

Sintaxis:

```
BIC-URI ::= "urn:epc:id:bic:" BICURIBody

BICURIBody ::= OwnerCode EquipCatId SerialNumber CheckDigit

OwnerCode ::= OnwerCodeChar*3

EquipCatId ::= CatIdChar*1

SerialNumber ::= Digit*6

CheckDigit ::= Digit
```

```
OwnerCodeChar ::= "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "J" | "K"
| "L" | "M" | "N" | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" |
"Y" | "Z"

CatIdChar ::= "J" | "U" | "Z"
```

6.4 Sintaxis del URI de clase de EPC

Esta sección especifica la sintaxis de URI de clase de EPC.

La sintaxis formal para el URI de clase de EPC es la siguiente:

```
EPCClass-URI ::= LGTIN-URI
```

donde las diversas alternativas en el lado derecho se especifican en las secciones que siguen.

Cada esquema de URI de clase de EPC se especifica en una de las siguientes subsecciones, de la siguiente manera:

Tabla 6-1 Esquemas de clase de EPC y donde se define la forma de identidad pura.

Esquema de Clase de EPC	Especificado en	Llave GS1 correspondiente	Uso típico
lgtin	Sección 6.4.1	GTIN + número de lote	Clase de objetos pertenecientes a un lote determinado

6.4.1 GTIN + lote (LGTIN)

El esquema de GTIN + lote se usa para denotar una clase de objetos que pertenecen a un determinado lote de un GTIN dado.

Sintaxis general:

`urn:epc:class:lgtin:`*`CompanyPrefix.ItemRefAndIndicator.Lot`*

Ejemplo:

```
urn:epc:class:lgtin:4012345.012345.998877
```

Sintaxis:

```
LGTIN-URI ::= "urn:epc:class:lgtin:" LGTINURIBody

LGTINURIBody ::= 2*(PaddedNumericComponent ".") GS3A3Component
```

La cantidad de caracteres en los dos campos `PaddedNumericComponent` debe sumar 13 (sin incluir ninguno de los caracteres de punto).

El campo de Lote del LGTIN-URI se expresa como un `GS3A3Component`, que permite la representación de todos los caracteres permitidos en el Número de lote del Identificador de Aplicación (10) de acuerdo con las Especificaciones Generales de GS1.

El LGTIN consta de los siguientes elementos:

- El **Prefijo GS1 de empresa**, asignado por GS1 a una entidad administradora o sus delegados. Es lo mismo que los dígitos del Prefijo GS1 de empresa dentro de una llave GTIN GS1. Consulte la Sección *7.3.2* para ver el caso de un GTIN-8.
- La **Referencia e Indicador del Artículo**, asignada por la entidad gestora a una clase de objeto en particular. La Referencia e Indicador del Artículo tal como aparece en el URI de EPC se obtiene del GTIN concatenando el dígito indicador del GTIN (o un carácter de relleno cero, si el URI de EPC se deriva de un GTIN-8, GTIN-12 o GTIN-13) con los dígitos de Referencia del Artículo y tratando el resultado como una única cadena numérica. Consulte la Sección *7.3.2* para ver el caso de un GTIN-8.

- El **Número de lote**, asignado por la entidad administradora a un lote distinto de una clase de objetos. El número de lote no es parte del GTIN, pero se usa para distinguir grupos individuales de la misma clase de objetos entre sí.

7 Correspondencia entre los EPC y las llaves GS1

Como se analizó en la Sección 4.3, existe una relación bien definida entre los Códigos Electrónicos de Productos (EPC) y siete llaves (más el identificador de componente/pieza) definidas en las Especificaciones Generales de GS1 [GS1GS]. Esta sección especifica la correspondencia entre los EPC y las llaves GS1.

7.1 El Prefijo GS1 de empresa (GCP) en las codificaciones de EPC

La correspondencia entre los EPC y las llaves GS1 se basa en identificar la parte de una llave GS1 que es el Prefijo GS1 de empresa. El Prefijo GS1 de empresa (GCP) es un número de 4 a 12 dígitos asignado por una Organización Miembro de GS1 a una entidad administradora, y la entidad administradora es libre de crear llaves GS1 usando ese GCP. Para los fines del Estándar de Datos de Etiquetas EPC, un GCP de 4 o 5 dígitos se trata como un bloque de 100 GCP de 6 dígitos o un bloque de 10 GCP de 6 dígitos, respectivamente. En el URI de EPC, el GCP está codificado en el componente ***CompanyPrefix***, que DEBE incluir el GCP de 4 o 5 dígitos y los siguientes 2 o 1 dígitos de la llave GS1, como si fuera un GCP de 6 dígitos. Luego, este valor se codifica en las codificaciones binarias de EPC utilizando el valor de partición 6 (binario: 110).

7.2 Determinación de la longitud del componente CompanyPrefix del EPC para las llaves GS1 asignadas de manera individual

En algunos casos, una Organización Miembro de GS1 asigna una llave GS1 asignada de manera individual (también conocida como “emisión única” o “única”), como un GTIN, GLN u otra llave completa, a una organización suscriptora. En tales casos, una organización suscriptora NO DEBE usar los dígitos que comprenden una llave particular asignada de manera individual para construir cualquier otro tipo de llave GS1. Por ejemplo, si una organización suscriptora recibe un GLN asignado de manera individual, NO DEBE crear SSCC con los 12 dígitos del GLN asignado de forma individual como si fuera un Prefijo GS1 de empresa de 12 dígitos.

Tenga en cuenta que una llave asignada de manera individual por lo general se resolverá (por ejemplo, a través de GEPIR) de vuelta a la MO emisora (ya que la MO se asignó el GCP en cuestión a sí misma con el fin de generar llaves asignadas de manera individual), en lugar de a la organización a la que se emitió la llave. La asignación de llaves asignadas de manera individual, basadas en un GCP común, a organizaciones suscritas dispares que no tienen una relación particular entre sí, es eficaz para prevenir el uso del componente ***CompanyPrefix*** de las codificaciones de EPC con fines de filtrado/correlación/consulta al nivel de una organización individual.

7.2.1 GTIN asignados de manera individual

Al codificar un GTIN asignado de manera individual como EPC, el GTIN-12, GTIN-13 o GTIN-8 emitido por la MO debe convertirse primero a un número de 14 dígitos anteponiendo dos, uno o seis ceros a la izquierda, respectivamente, al GTIN asignado de manera individual, como se especifica en las secciones 7.3.1 y 7.3.2.

Al GTIN asignado de forma individual, después aplicar de cualquier relleno necesario para aumentar su longitud a 14 dígitos, se le quita su dígito de verificación (que se omite en todas las codificaciones de EPC) y el dígito indicador o cero antepuesto, y se DEBE incluir en el componente ***CompanyPrefix*** del EPC, cuya longitud DEBE fijarse en 12 dígitos para un GTIN asignado de manera individual. Para un GTIN-12, GTIN-13 o GTIN-8, el componente *ItemRefAndIndicator* del EPC SGTIN resultante es un solo dígito cero. Para un GTIN-14, el componente *ItemRefAndIndicator* del EPC SGTIN resultante consiste en el cero a la izquierda o dígito indicador del GTIN-14.

Tenga en cuenta que estas reglas también aplican para los GTIN asignados de manera individual por terceros con el permiso de GS1.

Sintaxis:

urn:epc:id:sgtin:*CompanyPrefix.ItemRefAndIndicator.SerialNumber*

Ejemplo:

Cadena de elementos GS1: (01) 1234567890128 (21) 4711

URI de EPC: urn:epc:id:sgtin:123456789012.0.4711

La codificación binaria de EPC correspondiente (SGTIN-96 y SGTIN-198) usa el valor de partición 0, conforme a la Tabla 14-2 (*Tabla de particiones de SGTIN*).

7.2.2 GLN asignados de manera individual

Al codificar un GLN asignado de manera individual como EPC, todo el GLN asignado de manera individual (despojado de su dígito de verificación, que se omite de las codificaciones de EPC) ocupa el componente *CompanyPrefix* del EPC, cuya longitud se establece en 12 dígitos.

Para el EPC SGLN resultante, el componente *LocationReference* es una cadena de longitud cero. El componente *Extension* del EPC SGLN refleja el valor del componente de extensión de GLN, AI (254); si la cadena de elementos GS1 de entrada no incluyó un componente de extensión GLN (AI 254), el componente *Extension* del EPC SGLN comprende un solo dígito cero ('0').

Tenga en cuenta que estas reglas también se aplican a los GLN asignados de manera individual (por ejemplo, números comerciales nacionales) asignados por terceros con el permiso de GS1.

Sintaxis:

urn:epc:id:sgln:*CompanyPrefix*..*Extension*

Ejemplo (sin extensión):

Cadena de elementos GS1: (414) 1234567890128

EPC URI: urn:epc:id:sgln:123456789012..0

Ejemplo (con extensión):

Cadena de elementos GS1: (414) 1234567890128 (254) 4711

URI de EPC: urn:epc:id:sgln:123456789012..4711

La codificación binaria de EPC correspondiente (SGLN-96 y SGLN-195) usa el valor de partición 0, según la tabla 14-7 (***Tabla de particiones de SGLN***).

7.2.3 Otras llaves GS1 asignadas de manera individual

Otras llaves GS1 asignadas de manera individual (por ejemplo, SSCC, GIAI) deben codificarse como EPC con componentes ***CompanyPrefix*** de 12 dígitos de longitud.

En tales casos, una organización suscriptora NO DEBE usar los dígitos que comprenden una llave asignada de manera individual en particular para construir cualquier otra llave GS1. Por ejemplo, si una organización suscriptora recibe un SSCC asignado de manera individual, NO DEBE crear un SSCC adicional utilizando los 12 dígitos del SSCC asignado de manera individual como si fuera un GCP de 12 dígitos.

Ejemplo (SSCC):

Cadena de elementos GS1: (00) 012345678901234560

URI de EPC: urn:epc:id:sscc:123456789012.03456

Ejemplo (GIAI):

Cadena de elementos GS1: (8004) 12345678901234567890 1234567890

URI de EPC: urn:epc:id:giai:123456789012.34567890 1234567890

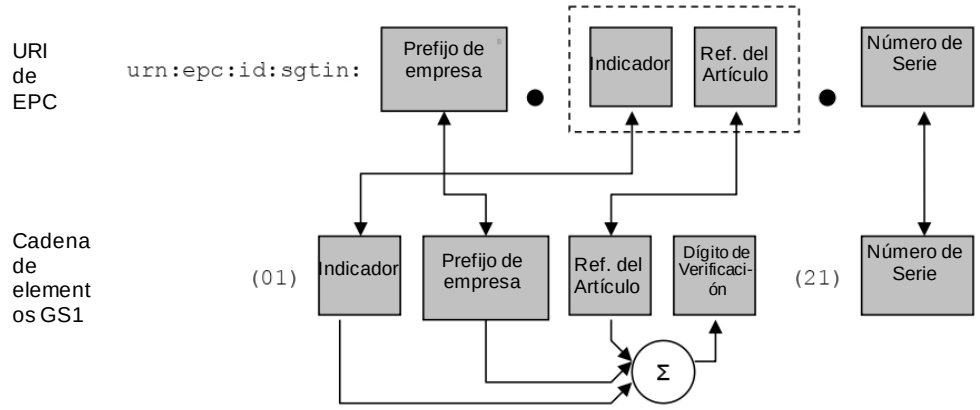
La codificación binaria de EPC correspondiente utiliza el valor de partición 0, según la Tabla de particiones respectiva en la sección 14.

7.3 Número Global de Artículo Comercial Serializado (SGTIN)

El EPC SGTIN (Sección *6.3.1*) no corresponde directamente a ninguna llave GS1, sino que corresponde a una combinación de una llave GTIN más un número de serie. El número de serie en el SGTIN se define como equivalente a AI 21 en las Especificaciones Generales de GS1.

La siguiente es una representación gráfica de la correspondencia entre el URI de EPC SGTIN y una cadena de elementos GS1 que consiste en una llave GTIN (AI 01) y un número de serie (AI 21):

Figura 7-1 Correspondencia entre el URI de EPC SGTIN y la cadena de elementos GS1.



(Tenga en cuenta que en el caso de un GTIN-12 o GTIN-13, un carácter de relleno cero ocupa el lugar del dígito indicador en la figura anterior.)

Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:sgtin: d1d2...d(L+i) . d1d(L+2) d(L+3).d13 . S1S2...SK`

Cadena de elementos GS1: `(01) d1d2...d14 (21)` $S_1S_2...SK$

donde $1 \leq K \leq 20$.

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC SGTIN:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 13 dígitos.
2. Numere los caracteres del componente de número de serie (tercero) del EPC como se muestra antes. Cada s_i corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación $d14 = (10 - ((3(d1 + d3 + d5 + d7 + d9 + d11 + d13) + (d2 + d4 + d6 + d8 + d10 + d12)) \bmod 10)) \bmod 10$.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier s_i en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GTIN (AI 01) y un número de serie (AI 21):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Salvo en el caso de un GTIN-8, determine la cantidad de dígitos *L* en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa. Consulte la Sección 7.3.2 para ver el caso de un GTIN-8.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GTIN d_{14} no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente en la columna "Forma de URI" de la Tabla A-1, ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal.

   Ejemplo:

   URI de EPC: `urn:epc:id:sgtin:0614141.712345.32a%2Fb`

   Cadena de elementos GS1: `(01) 7 0614141 12345 1 (21) 32a/b`

   Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero normalmente no están presentes. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.3.1 GTIN-12 y GTIN-13

Para encontrar el URI de EPC correspondiente a la combinación de un GTIN-12 o GTIN-13 y un número de serie, primero convierta el GTIN-12 o GTIN-13 en un número de 14 dígitos agregando dos o un cero a la izquierda, respectivamente, como se muestra en [GS1GS19.0], Sección 3.3.2.

Ejemplo:

GTIN-12: 2 12345 614141

Número correspondiente de 14 dígitos: 0 0614141 12345 2

EPC SGTIN correspondiente: urn:epc:id:sgtin:0614141.012345.Serial

Ejemplo:

GTIN-13: 2 12345 0614141

Número correspondiente de 14 dígitos: 0 0614141 12345 2

EPC SGTIN correspondiente: `urn:epc:id:sgtin:0614141.012345.Serial`

En estos ejemplos se han agregado espacios a las cadenas de GTIN para mayor claridad, pero nunca se codifican.

7.3.2 GTIN-8

Un GTIN-8 es un caso especial del GTIN que se utiliza para identificar artículos comerciales pequeños.

El código GTIN-8 consta de ocho dígitos N_1, $N_2 \ldots N_8$, donde los primeros dígitos N_1 a N_L son el Prefijo GS1-8 (donde L = 1, 2 o 3), los siguientes dígitos N_{L+1} a N_7 son la Referencia del Artículo y el último dígito N_8 es el dígito de verificación. El Prefijo GS1-8 es un número de índice de uno, dos o tres dígitos, administrado por la Oficina Global GS1. No identifica el origen del artículo. La Referencia del Artículo es asignada por la Organización Miembro de GS1. Las Organizaciones Miembro de GS1 brindan los procedimientos para obtener los GTIN-8.

Para encontrar el URI de EPC correspondiente a la combinación de un GTIN-8 y un número de serie, se DEBE utilizar el siguiente procedimiento. Para el procedimiento definido anteriormente en la Sección 7.1, la porción del Prefijo GS1 de empresa del EPC se construirá anteponiendo cinco ceros a los primeros tres dígitos del GTIN-8; es decir, la porción del Prefijo GS1 de empresa del EPC tiene ocho dígitos y será de $00000N_1N_2N_3$.

La Referencia del Artículo para el procedimiento serán los dígitos restantes del GTIN-8 además del dígito de verificación, es decir, N_4 a N_7. El Dígito Indicador del procedimiento será cero.

Ejemplo:

GTIN-8: 95010939

EPC SGTIN correspondiente: `urn:epc:id:sgtin:00000950.01093.`*`Serial`*

7.3.3 RCN-8

Un RCN-8 es un código de 8 dígitos que comienza con los Prefijos GS1-8 0 o 2, como se define en [GS1GS19.0], Sección 2.1.11.1. Estos están reservados para la numeración interna de la empresa y no son códigos GTIN-8. Los códigos RCN-8 NO DEBEN utilizarse para construir EPC SGTIN y el procedimiento para los códigos GTN-8 no es aplicable.

7.3.4 Numeración interna de la empresa (Prefijos GS1 04 y 0001 - 0007)

Las Especificaciones Generales de GS1 reservan códigos que comienzan con 04 o 0001 hasta 0007 para la numeración interna de la empresa. (Véase [GS1GS19.0], Secciones 2.1.11.2 y 2.1.11.3.)

Estos números NO DEBEN usarse para construir EPC SGTIN. Una versión futura del Estándar de Datos de Etiquetas de EPCglobal puede especificar reglas normativas para el uso de códigos de Numeración Interna de la empresa en los EPC.

7.3.5 Circulación restringida (Prefijos GS1 02 y 20-29)

Las Especificaciones Generales de GS1 reservan códigos que comienzan con 02 o 20 hasta 29 para su circulación restringida en áreas geopolíticas definidas por organizaciones miembros de GS1 y para artículos comerciales de medida variable. (Véase [GS1GS19.0], Secciones 2.1.11.1 y 2.1.11.14.)

Estos números NO DEBEN usarse para construir EPC SGTIN. Una versión futura del Estándar de Datos de Etiquetas de EPCglobal puede especificar reglas normativas para el uso de códigos de Circulación Restringida en los EPC.

7.3.6 Identificación del código de cupón para distribución restringida (Prefijos GS1 981-984 y 99)

Los cupones se pueden identificar mediante códigos construidos de acuerdo con las Secciones 2.6.1-2.6.3 de las Especificaciones Generales de GS1. Los números resultantes comienzan con los Prefijos GS1 981-984 y 99. Sin embargo, estrictamente hablando, un cupón no es un artículo comercial y estos códigos de cupón no son en realidad números de identificación de artículo comercial.

Por lo tanto, los códigos de cupón para distribución restringida NO DEBEN usarse para construir EPC SGTIN.

7.3.7 Recibo de reembolso (Prefijo GS1 980)

La Sección 2.6.4 de la Especificación General de GS1 especifica la construcción de códigos para representar los recibos de reembolso, como los creados por las máquinas de reciclaje de botellas para el canje en el punto de venta. El número resultante comienza con el Prefijo GS1 980. Sin embargo, estrictamente hablando, un recibo de reembolso no es un artículo comercial y estos códigos de recibo de reembolso no son en realidad números de identificación de artículo comercial.

Por lo tanto, los códigos de recibo de reembolso NO DEBEN usarse para construir EPC SGTIN.

7.3.8 ISBN, ISMN e ISSN (Prefijos GS1 977, 978 o 979)

Las Especificaciones Generales de GS1 prevén el uso de un identificador de 13 dígitos para representar los códigos ISBN (Número Internacional Normalizado del Libro), ISMN (Número Internacional Normalizado para Música) e ISSN (Número Internacional Normalizado de Publicaciones Seriadas). El código resultante es un GTIN cuyo Prefijo GS1 es 977, 978 o 979.

7.3.8.1 ISBN e ISMN

Los códigos ISBN e ISMN se utilizan para libros y música impresos, respectivamente. Los códigos están definidos por normas ISO (ISO 2108 para el ISBN e ISO 10957 para el ISMN) y los administra la Agencia Internacional del ISBN(*http://www.isbn-international.org/*) y las agencias de registro nacionales afiliadas. El ISMN es de una organización distinta (*http://www.ismn-international.org/*) pero su gestión y la estructura de su codificación son similares a las del ISBN.

Si bien estos códigos no los asigna GS1, tienen una estructura interna muy similar que se presta fácilmente a un tratamiento similar al momento de crear EPC. Un código ISBN consta de las siguientes partes, que se muestran a continuación con el concepto correspondiente del sistema GS1:

Elemento de Prefijo + Elemento de Grupo del Registrante	= Prefijo GS1 (978 o 979 más dígitos adicionales)
Elemento del Registrante	= Resto del Prefijo GS1 de empresa
Elemento de Publicación	= Referencia del Artículo
Digito de verificación	= Digito de verificación

Los Elementos de Grupo del Registrante se asignan a las agencias de registro de ISBN, que a su vez asignan Elementos del Registrante a los editores, quienes asignan Elementos de Publicación a las ediciones de publicaciones individuales. Esto es exactamente paralelo a la construcción de los códigos GTIN. Al igual que en el GTIN, los diversos componentes son de longitud variable y, al igual que en el GTIN, cada editor conoce la longitud combinada del Elemento de Grupo del Registrante y el Elemento de Registrante, ya que la combinación se asigna al editor. La longitud total del Elemento de Prefijo "978" o "979", el Elemento de Grupo del Registrante y el Elemento del Registrante está en el rango de 6 a 12 dígitos, que es exactamente el rango de longitudes de Prefijo GS1 de empresa permitidas en el EPC SGTIN. De esta manera, el ISBN y el ISMN se pueden utilizar para construir SGTIN como se especifica en esta norma.

Para encontrar el URI de EPC correspondiente a la combinación de un ISBN o ISMN y un número de serie, se DEBE utilizar el siguiente procedimiento. Para el procedimiento definido anteriormente en la Sección *7.1*, la porción de Prefijo GS1 de empresa del EPC se deberá construir concatenando el Elemento de Prefijo ISBN/ISMN (978 o 979), el Elemento de Grupo del Registrante y el Elemento del Registrante. La Referencia del Artículo para el procedimiento serán los dígitos del Elemento de Publicación ISBN/ISMN. El Dígito Indicador del procedimiento será cero.

Ejemplo:

ISBN: 978-81-7525-766-5

EPC SGTIN correspondiente: `urn:epc:id:sgtin:978817525.0766.Serial`

7.3.8.2 ISSN

El ISSN es el código normalizado internacional que permite la identificación de cualquier publicación seriada, incluidas las seriadas electrónicas, independientemente de su país de publicación, de su idioma o alfabeto, de su frecuencia, medio, etc. El código está definido por la norma ISO (ISO 3297) y lo administra la Agencia Internacional de ISSN (*http://www.issn.org/*).

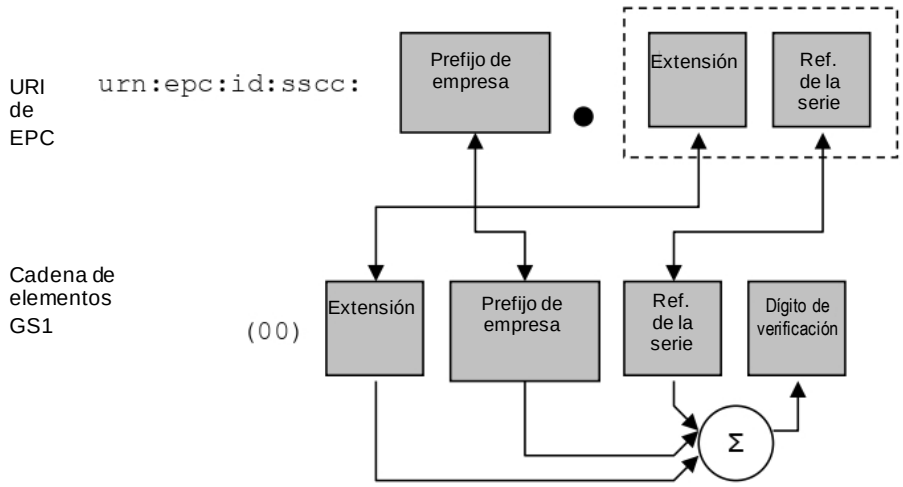
El ISSN es un GTIN que comienza con el prefijo GS1 977. La estructura del ISSN no permite que se exprese en formato SGTIN. Por lo tanto, en espera de los requisitos formales que surjan del sector de las publicaciones seriadas, actualmente no es posible crear un SGTIN con base en un ISSN.

7.4 Código Seriado de Contenedor de Envío (SSCC)

El EPC SSCC (Sección *6.3.2*) corresponde directamente a la llave SSCC definida en las Secciones 2.2.1 y de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC SSCC y una cadena de elementos GS1 que consiste en una llave SSCC (AI 00):

Figura 7-2 Correspondencia entre el URI de EPC SSCC y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id: sscc: d2d3...d(L+1) . did(L+2)d(L+3)...d17`

Cadena de elementos GS1: `(00) d1d2...d18`

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC SSCC:

1. Numere los dígitos de los dos componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 17 dígitos.
2. Calcule el dígito de verificación d18 = (10 - ((3(d1 + d3 + d5 + d7 + d9 + d11 + d13 + d15 + d17) + (d2 + d4 + d6 + d8 + d10 + d12 + d14 + d16)) mod 10)) mod 10.
3. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un SSCC (AI 00):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos L en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación SSCC d18 no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: urn:epc:`id:sscc:0614141.1234567890`

Cadena de elementos GS1: `(00) 1 0614141 234567890 8`

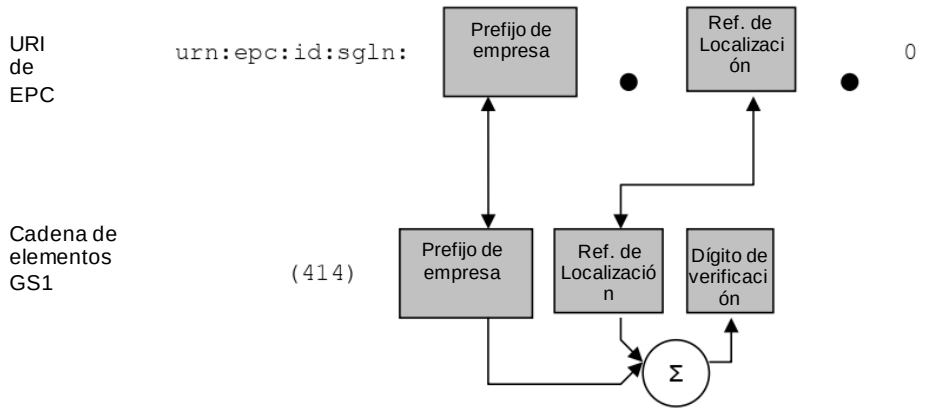
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.5 Número Global de Localización con o sin Extensión (SGLN)

El EPC SGLN (Sección *6.3.3*) corresponde directamente a una llave Número Global de Localización (GLN), como se especifica en las Secciones 2.4.4 y 3.7.9 de las Especificaciones Generales de GS1 [GS1GS19.0], o a la combinación de un GLN más un número de extensión como se especifica en la Sección 3.5.11 de [GS1GS19.0]. Se reserva un número de extensión cero para indicar que un EPC SGLN denota un GLN no extendido, en lugar de una extensión GLN más. (Véase la Sección *6.3.3* para obtener una explicación de la letra "S" en "SGLN").

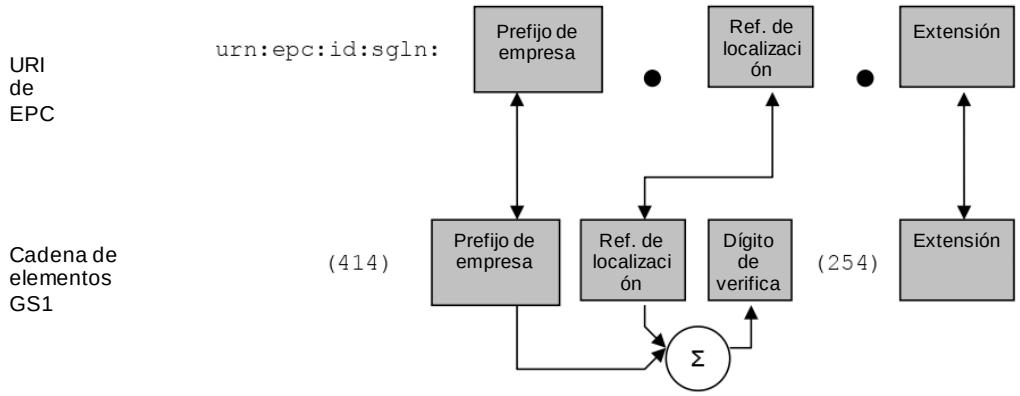
A continuación, se muestra gráficamente la correspondencia entre el URI de EPC SGLN y una cadena de elementos GS1 que consiste en una llave GLN (AI 414) ***sin*** una extensión:

Figura 7-3 Correspondencia entre el URI de EPC SGLN sin extensión y la cadena de elementos GS1



A continuación, se muestra gráficamente la correspondencia entre un URI de EPC SGLN y una cadena de elementos GS1 que consiste en una llave GLN (AI 414) junto con una extensión (AI 254):

Figura 7-4 Correspondencia entre el URI de EPC SGLN con extensión y la cadena de elementos GS1



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:sgln: d1d2...dL . d(L+1)d(L+2)-d12 . S1S2-SK`

Cadena de elementos GS1: `(414) d1d2-d13 (254) S1S2-SK`

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC SGLN:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 12 dígitos.
2. Numere los caracteres del componente *Extensión* (tercero) del EPC como se muestra antes. Cada s_i corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación $d_{13} = (10 - ((3(d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12}) + (d_1 + d_3 + d_5 + d_7 + d_9 + d_{11})) \bmod 10)) \bmod 10$.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier s_1 en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Si el número de serie consta de un solo carácter s_i y ese carácter es el dígito cero ('0'), omita la extensión de la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GLN (AI 414) con y sin una extensión adjunta (AI 254):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GLN d_{13} no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente **en la columna** "Forma de URI" de la *Tabla A-1*, ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal. Si la cadena de elementos GS1 de entrada no incluía una extensión (AI 254), use un solo dígito cero ('0') como el número de serie completo $_{S1, S2...SK}$ en el URI de EPC.

Ejemplo (sin extensión):

URI de EPC: `urn:epc:id:sgln:0614141.12345.0`

Cadena de elementos `GS1: (414) 0614141 12345 2`

Ejemplo (con extensión):

URI de EPC: `urn:epc:id:sgln:0614141.12345.32a%2Fb`

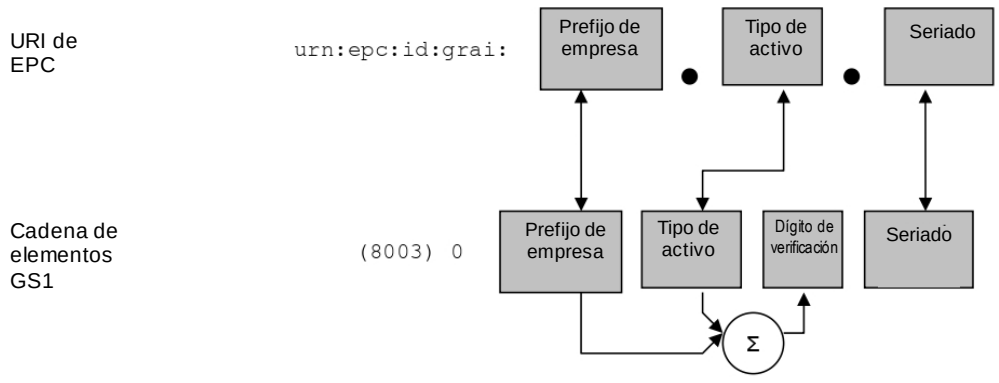
Cadena de elementos `GS1: (414) 0614141 12345 2 (254) 32a/b`

Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.6 Identificador Global de Activos Retornables (GRAI)

El EPC GRAI (Sección 6.3.4) corresponde directamente a la llave GRAI serializada definida en las Secciones 2.3 y 3.9.3 de las Especificaciones Generales de GS1 [GS1GS19.0]. Debido a que un EPC siempre identifica un objeto físico específico, solo las llaves GRAI que incluyen el número de serie opcional tienen un EPC GRAI correspondiente. Las llaves GRAI que carecen de un número de serie se refieren a clases de activos en lugar de activos específicos y, por lo tanto, no tienen un EPC correspondiente (de la misma manera en que una llave GTIN sin un número de serie no tiene un EPC correspondiente).

Figura 7-5 Correspondencia entre el URI de EPC GRAI y la cadena de elementos GS1.



Tenga en cuenta que la cadena de elementos GS1 incluye un dígito cero (`'0'`) adicional después del Identificador de Aplicación (8003). Este dígito cero es un relleno adicional en la cadena de elementos y *no* forma parte de la llave GRAI en sí.

Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: urn:epc:id:grai:did2. dL.d(L+i)d(L+2). di2.sis2. sK

Cadena de elementos GS1: (8003)0did2. di3sis2. sK

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GRAI:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 12 dígitos.
2. Numere los caracteres del componente de número de serie (tercero) del EPC como se muestra antes. Cada s_i corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación d_{13} = (10 - ((3(d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12}) + (d_1 + d_3 + d_5 + d_7 + d_9 + d_{10})) mod 10)) mod 10.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier s_i en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GRAI (AI 8003):

1. Si el número de caracteres que sigue al identificador de la aplicación (8003) es igual o menor que 14, deténgase: esta cadena de elementos no tiene un EPC correspondiente porque no incluye el número de serie opcional.
2. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
3. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
4. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GRAI $\boldsymbol{d}_{13}$ no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente en la columna "Forma de URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal.

Ejemplo:

URI de EPC: urn:epc:id:grai:0614141.12345.32a%2Fb

Cadena de elementos GS1: (8003) 0 0614141 12345 2 32a/b

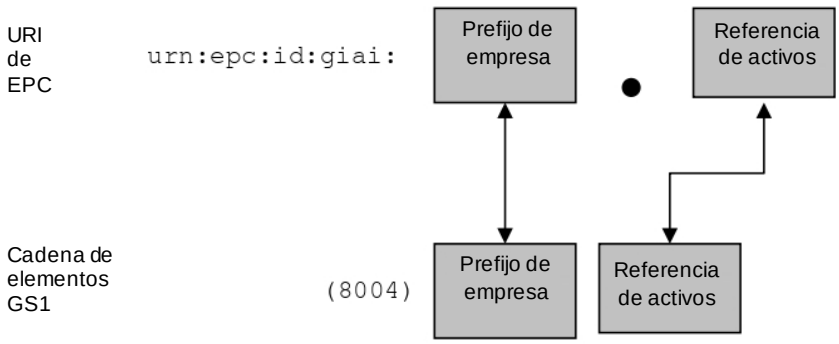
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.7 Identificador Global Individual de Activo (GIAI)

El EPC GIAI (Sección *6.3.5*) corresponde directamente a la llave GIAI definida en las Secciones 2.3.2 y 3.9.4 de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC GIAI y una cadena de elementos GS1 que consiste en una llave GIAI (AI 8004):

Figura 7-6 Correspondencia entre el URI de EPC GIAI y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:giai: d1d2...dL. S1S2.SK`

Cadena de elementos GS1: `(8004) d1d2.dLS1s2.sK`

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GIAI:

1. Numere los caracteres de los dos componentes del EPC como se muestra antes. Cada *si* corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
2. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier *si* en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GIAI (AI 8004):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Reemplace cada carácter de número de serie *si* con el valor correspondiente en la columna "Forma de URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete de escape porcentual si *si* no es un carácter URI legal.

   URI de EPC: `urn:epc:id:giai:0614141.32a%2Fb`

   Cadena de elementos GS1: `(8004) 0614141 32a/b`

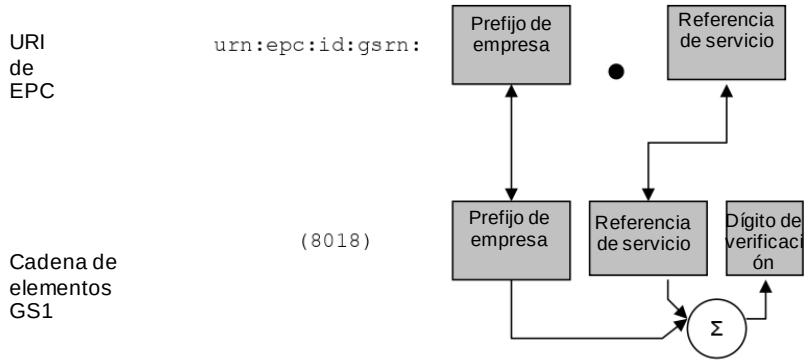
   Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.8 Número Global de Relación del Servicio - Receptor (GSRN)

El EPC GSRN (Sección *6.3.6*) corresponde directamente a la llave GSRN - Receptor definida en las Secciones 2.5.2 y 3.9.14 de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC GSRN y una cadena de elementos GS1 que consiste en una llave GSRN (AI 8018):

Figura 7-7 Correspondencia entre el URI de EPC GSRN y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:gsrn: d1d2...dL. d(L+1)d(L+2)…d17`

Cadena de elementos GS1: `(8018) d1d2...d18`

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GSRN:

1. Numere los dígitos de los dos componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 17 dígitos.
2. Calcule el dígito de verificación $\boldsymbol{d}_{18} = (10 - ((3(d_1 + d_3 + d_5 + d_7 + d_9 + d_{11} + d_{13} + d_{15} + d_{17}) + (d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12} + d_{14} + d_{16})) \bmod 10)) \bmod 10$.
3. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GSRN - Receptor (AI 8018):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GSRN $\boldsymbol{d}_{18}$ no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: urn:epc:id:`gsrn:0614141.1234567890`

Cadena de elementos GS1: `(8018) 0614141 1234567890 2`

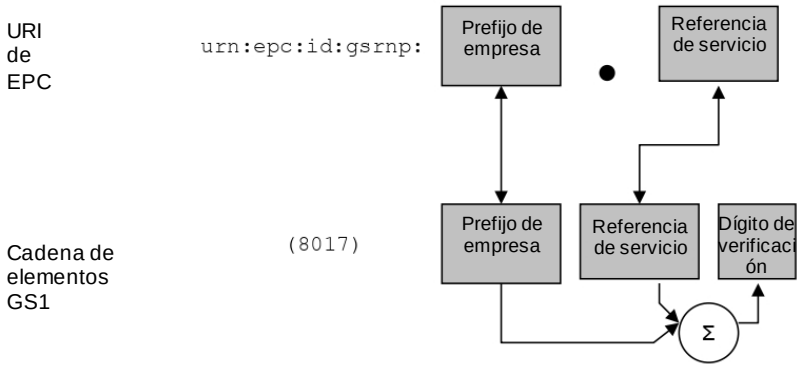
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.9 Número Global de Relación del Servicio - Prestador (GSRNP)

El EPC GSRNP (Sección *6.3.6*) corresponde directamente a la llave GSRN - Proveedor definida en las Secciones 2.5.1 y 3.9.14 de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC GSRNP y una cadena de elementos GS1 que consiste en una llave GSRN - Proveedor (AI 8017):

Figura 7-8 Correspondencia entre el URI de EPC GSRNP y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: *urn:epc:id:gsrnp:d1d2…dL.d(L+1)d(L+2)…d17*

Cadena de elementos GS1: *(8017) d1d2….d18*

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GSRNP:

1. Numere los dígitos de los dos componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 17 dígitos.
2. Calcule el dígito de verificación d_{18} = (10 - ((3(d_1 + d_3 + d_5 + d_7 + d_9 + d_{11} + d_{13} + d_{15} + d_{17}) + (d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12} + d_{14} + d_{16})) mod 10)) mod 10.
3. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GSRN - Proveedor (AI 8017):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GSRN d_{18} no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: `urn:epc:id:gsrnp:0614141.1234567890`

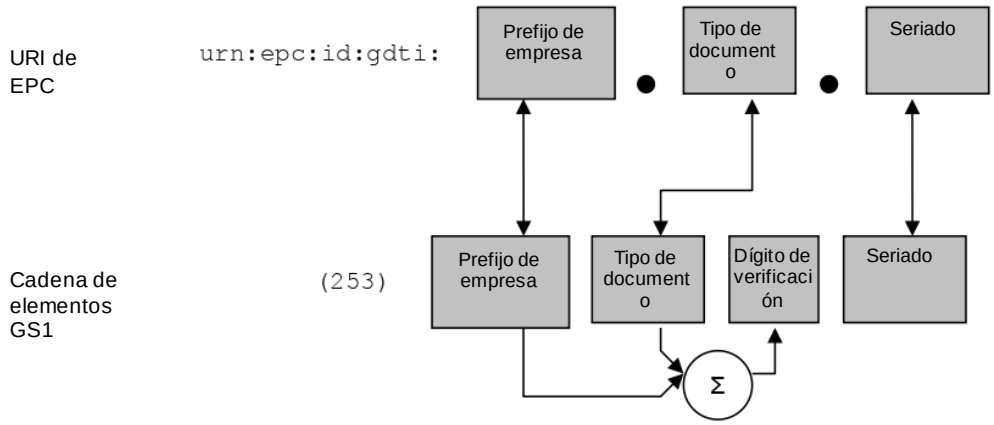
Cadena de elementos GS1: `(8017) 0614141 1234567890 2`

Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.10 Identificador de tipo de documento global (GDTI)

El EPC GDTI (Sección *6.3.7*) corresponde directamente a la llave GDTI serializada definida en las Secciones 2.6.9 y 3.5.10 de las Especificaciones Generales de GS1 [GS1GS19.0]. Debido a que un EPC siempre identifica un objeto físico específico, solo las llaves GDTI que incluyen el número de serie opcional tienen un EPC GDTI correspondiente. Las llaves GDTI que carecen de un número de serie se refieren a clases de documentos en lugar de documentos específicos y, por lo tanto, no tienen un EPC correspondiente (al igual que una llave GTIN sin un número de serie no tiene un EPC correspondiente).

Figura 7-9 Correspondencia entre el URI de EPC GDTI y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:gdti:` $d_1d_2\ldots d_L$. $d_{(L+1)}d_{(L+2)}\ldots d_{12}$. $S_1S_2\ldots S_K$

Cadena de elementos GS1: `(253)` d1d2…d13s1s2…sK.

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GDTI:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 12 dígitos.
2. Numere los caracteres del componente de número de serie (tercero) del EPC como se muestra antes. Cada s_i corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación d_{13} = (10 - ((3(d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12}) + (d_1 + d_3 + d_5 + d_7 + d_9 + d_{11})) mod 10)) mod 10.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier s_i en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GDTI (AI 253):

1. Si el número de caracteres que sigue al identificador de la aplicación `(253)` es igual o menor que 13, deténgase: esta cadena de elementos no tiene un EPC correspondiente porque no incluye el número de serie opcional.
2. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
3. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
4. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GDTI d_{13} no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente en la columna "Forma de URI" de la *Tabla A-1*, ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal.

Ejemplo:

URI de EPC: `urn:epc:id:gdti:0614141.12345.006847`

Cadena de elementos GS1: `(253) 0614141 12345 2 006847`

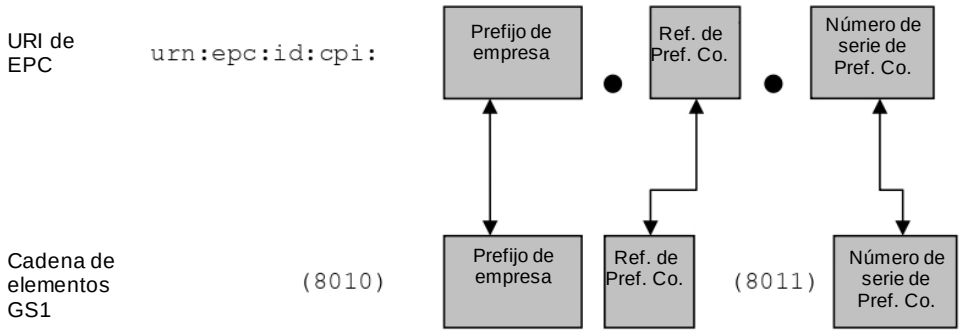
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.11 Identificador de Partes y Componentes (CPI)

El EPC CPI (Sección 6.3.9) no corresponde directamente a ninguna llave GS1, sino que corresponde a una combinación de dos elementos de datos definidos en las secciones 3.9.10 y 3.9.11 de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC CPI y una cadena de elementos GS1 que consiste en un Identificador de Componente / Pieza (AI 8010) y un número de serie de Componente / Pieza (AI 8011):

Figura 7-10 Correspondencia entre el URI de CPI EPC y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: *urn:epc:id:cpi: d1d2…dL. d(L+1)d(L+2)…dN. s1s2…sK*

Cadena de elementos GS1: *(8010) d1d2…dN (8011) s1s2…sK*

donde 1 ≤ N ≤ 30 y 1 ≤ K ≤ 12.

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC CPI:

1. Numere los dígitos de los tres componentes del EPC como se muestra antes. Cada *d*i en el segundo componente corresponde a un solo carácter o a un triplete de escape porcentual que consta de un carácter % seguido de dos dígitos hexadecimales.
2. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier *d*i en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla G-1* (*G*) (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla G-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye tanto un Identificador Componente / Pieza (AI 8010) como un Número de Serie Componente / Pieza (AI 8011):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos *L* en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Organice los caracteres como se muestra para el URI de EPC. Reemplace cada carácter de parte/componente *d*i con el valor correspondiente en la columna "Forma URI" de la *Tabla G-1* (*G*), ya sea el carácter en sí o un triplete de escape porcentual si *d*i no es un carácter URI legal.

Ejemplo:

URI de EPC: urn:epc:id:cpi:0614141.5PQ7%2FZ43.12345

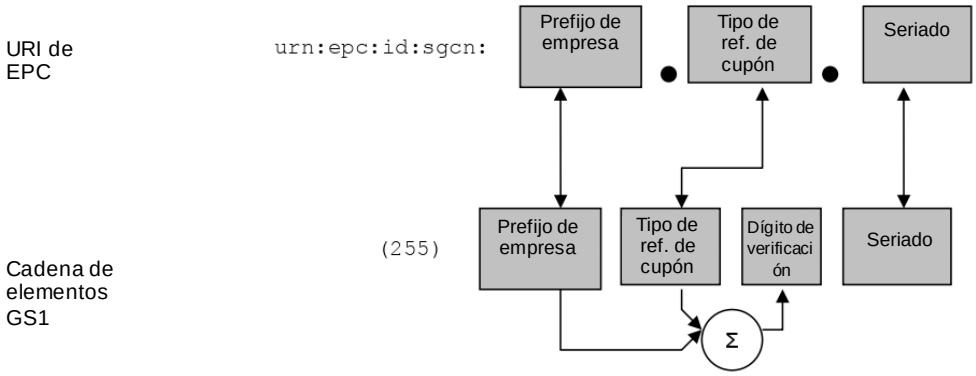
Cadena de elementos GS1: (8010) 0614141 5PQ7/Z43 (8011) 12345

Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero normalmente no están presentes. En este ejemplo, el carácter de barra diagonal (/) en la referencia de parte/componente debe representarse como un triplete de escape en el URI de EPC.

7.12 Número de Cupón Global Serializado (SGCN)

El EPC SGCN (Sección 6.3.10) corresponde directamente a la llave GCN serializada definida en las Secciones 2.6.1 y 3.5.12 de las Especificaciones Generales de GS1 [GS1GS19.0]. Debido a que un EPC siempre identifica un objeto físico o digital específico, solo las llaves SGCN que incluyen el número de serie tienen un EPC SGCN correspondiente. Las llaves GCN que carecen de un número de serie se refieren a clases de cupones en lugar de cupones específicos y, por lo tanto, no tienen un EPC correspondiente.

Figura 7-11 Correspondencia entre el URI de EPC SGCN y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: urn:epc:id:sgcn*: d1d2...dL . d(L+i)d(L+2)-di2 . S1S2...Sk*

Cadena de elementos GS1*: (255) d1-d2-...d13s1s2...sK*

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC SGCN:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 12 dígitos.
2. Numere los caracteres del componente de número de serie (tercero) del EPC como se muestra antes. Cada si es un carácter de dígito.
3. Calcule el dígito de verificación $d13 = (10 - ((3(d2 + d4 + d6 + d8 + d10 + d12) + (d1 + d3 + d5 + d7 + d9 + d11)) \bmod 10)) \bmod 10$.
4. Acomode los dígitos resultantes como se muestra para la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GCN (AI 255):

1. Si el número de caracteres que sigue al identificador de la aplicación (255) es igual o menor que 13, deténgase; esta cadena de elementos no tiene un EPC correspondiente porque no incluye el número de serie opcional.
2. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.

3. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.

4. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GCN ***d***13 no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: `urn:epc:id:sgcn:4012345.67890.04711`

Cadena de elementos GS1: `(255) 4012345 67890 1 04711`

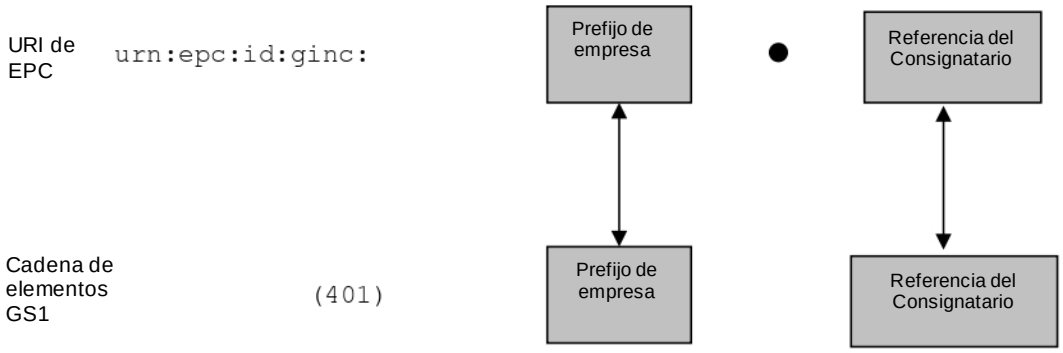
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.13 Número de Identificación Global para Consignación (GINC)

El EPC GINC (Sección 6.5.i) corresponde directamente a la llave GINC definida en las Secciones 2.2.2 y de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC GINC y una cadena de elementos GS1 que consiste en una llave GINC (AI 401):

Figura 7-12 Correspondencia entre el URI de EPC GINC y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:ginc:` *`did2…dL. s1s2…sK`*

Cadena de elementos GS1*: (401) d1d2…dLs1s2…sK*

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GINC:

1. Numere los caracteres de los dos componentes del EPC como se muestra antes. Cada si corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.

2. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier si en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GINC (AI 401):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.

2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.

3. Acomode los dígitos como se muestra para el URI de EPC. Reemplace cada carácter de número de serie si con el valor correspondiente en la columna "Forma URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete de escape porcentual si si no es un carácter URI legal.

Ejemplo:

URI de EPC: `urn:epc:id:ginc:0614141.xyz47%2F1`

Cadena de elementos GS1: `(401) 0614141 xyz47/11`

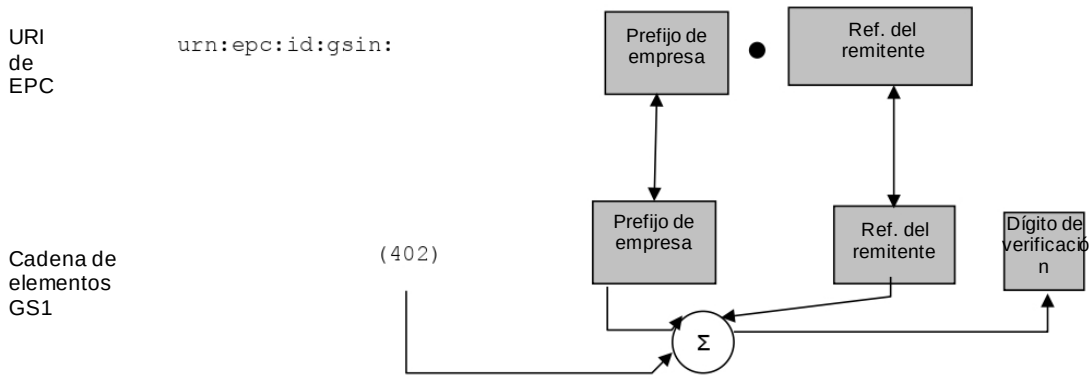
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.14 Número Global de Identificación de Envío (GS1N)

El EPC GS1N (Sección 6.5.2) corresponde directamente a la llave GS1N definida en las Secciones 2.2.3 y de las Especificaciones Generales de GS1 [GS1GS19.0].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC GS1N y una cadena de elementos GS1 que consiste en una llave GS1N (AI 402):

Figura 7-13 Correspondencia entre el URI de EPC GS1N y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: *urn:epc:id:gsin:d1d2…dL.d(L+1)d(L+2)d(L+3)…d16*

Cadena de elementos GS1: (*402) d1d2. D17*

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC GS1N:

1. Numere los dígitos de los dos componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 16 dígitos.
2. Calcule el dígito de verificación $d17 = (10 - (((d1 + d3 + d5 + d7 + d9 + d11 + d13 + d15) + 3(d2 + d4 + d6 + d8 + d10 + d12 + d14 + d16)) \bmod 10)) \bmod 10$.

Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1.

1. Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GS1N (AI 402):
2. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
3. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
4. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GS1N ***d***17 no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: `urn:epc:id:gsin:0614141.123456789`

Cadena de elementos GS1: `(402) 0614141 123456789 0`

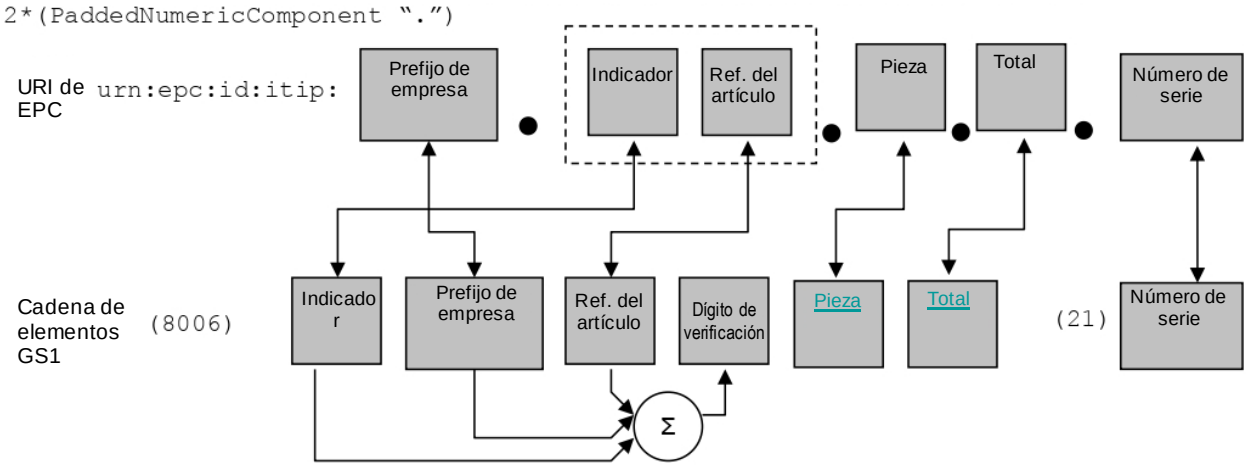
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero nunca se codifican.

7.15 Artículo Comercial Individual (ITIP)

El EPC (Sección 6.3.13) no corresponde directamente a ninguna llave GS1, sino que corresponde a una combinación de los AI (8006) y (21).

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC ITIP y una cadena de elementos GS1 que consiste en el AI (8006) y el AI (21):

Figura 7-14 Correspondencia entre el URI de EPC ITIP y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: *urn:epc:id:itip:d1d2-d(L+1) . did(L+2) d(L+3)...d13 .) . d1d2.d1d2.s1s2...SK*

Cadena de elementos GS1: *(8006) d1d2…d18 (21)s1s2…sK*

donde 1 ≤ K ≤ 20.

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC ITIP:

1. Numere los dígitos de los cuatro primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 17 dígitos.
2. Numere los caracteres del componente del número de serie (séptimo) del EPC como se muestra antes. Cada si corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación $d_{14} = (10 - ((3(d_1 + d_3 + d_5 + d_7 + d_9 + d_{11} + d_{13}) + (d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12})) \bmod 10)) \bmod 10$.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier si en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; en la columna "Símbolo gráfico" se proporciona el carácter correspondiente que se debe usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye tanto el AI (8006) como el AI (21):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.

2. Salvo en el caso de un GTIN-8, determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa. Consulte la Sección *7.1.2* para ver el caso de un GTIN-8.

3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GTIN d_{14} no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente en la columna "Forma de URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal.

Ejemplo:

URI de EPC: `urn:epc:id:itip:4012345.012345.04.04.32a%2Fb`

Cadena de elementos GS1: `(8006) 0 4012345 12345 6 04 04 (21) 32a/b`

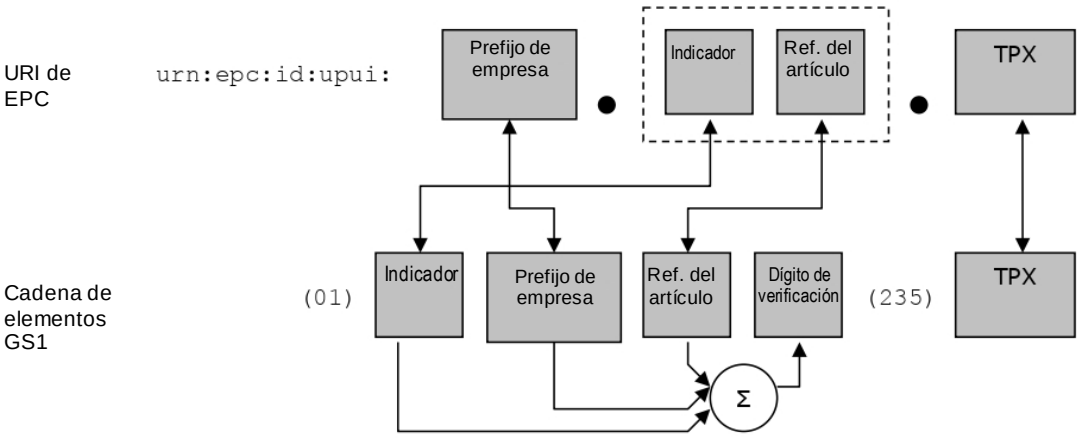
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero normalmente no están presentes. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de EPC.

7.16 Identificador a Nivel de las Unidades de Envasado (UPUI)

El EPC UPUI (Sección 6.3.14) no corresponde directamente a ninguna llave GS1, sino que corresponde a una combinación de una llave GTIN más una *Extensión serializada de GTIN controlada por terceros* (TPX), como se especifica en las Especificaciones Generales de GS1 [GS1GS].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC UPUI y una cadena de elementos GS1 que consiste en una llave GTIN (AI 01) y una *Extensión serializada de GTIN controlada por terceros* (AI 235):

Figura 7-15 Correspondencia entre el URI de EPC UPUI y la cadena de elementos GS1.



(Tenga en cuenta que en el caso de un GTIN-12 o GTIN-13, un carácter de relleno cero ocupa el lugar del dígito indicador en la figura anterior.)

Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: $urn:epc:id:upui:\ d_1d_2\ldots d_{(L+1)}.\ d_1d_{(L+2)}d_{(L+3)}\ldots d_{13}.\ s_1s_2.\ s_K$

Cadena de elementos GS1: $(01)\ d1d2\ldots d14\ (235)\ s1s2\ldots sK$

donde $1 \le K \le 28$.

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC UPUI:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 13 dígitos.

2. Numere los caracteres del tercer componente (TPX) del EPC como se muestra antes. Cada s_i corresponde a un solo carácter o a un triplete de escape porcentual que consiste en un carácter de % seguido de dos dígitos hexadecimales.

3. Calcule el dígito de verificación $d_{14} = (10 - ((3(d_1 + d_3 + d_5 + d_7 + d_9 + d_{11} + d_{13}) + (d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12})) \bmod 10)) \bmod 10$.

4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier s_i en el URI de EPC es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; la columna "Símbolo gráfico" proporciona el carácter correspondiente para usar en la cadena de elementos GS1).

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye tanto un GTIN (AI 01) como una *Extensión serializada controlada por terceros de GTIN* (AI 235):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.

2. Salvo en el caso de un GTIN-8, determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa. Consulte la Sección *7.1.2* para ver el caso de un GTIN-8.

3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GTIN d_{14} no se incluye en el URI de EPC. Reemplace cada carácter de número de serie s_i con el valor correspondiente en la columna "Forma de URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete de escape porcentual si s_i no es un carácter URI legal.

Ejemplo:

URI de EPC: `urn:epc:id:upui:1234567.089456.51qIgY)%3C%26Jp3*j`SDB`

Cadena de elementos GS1: `(01) 0 1234567 89456 0 (235) 51qIgY)<&Jp3*j7`SDB`

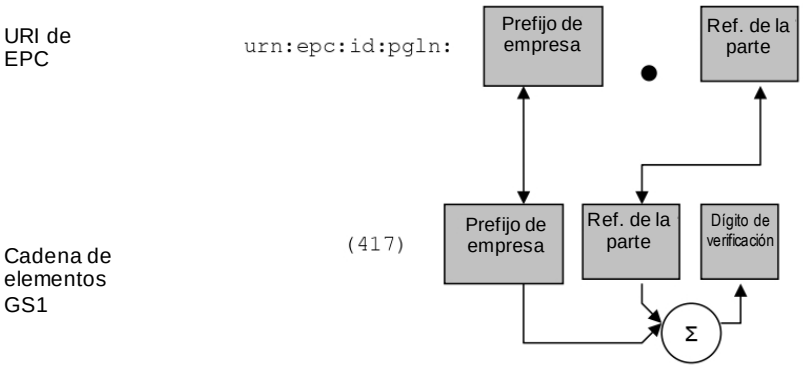
Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero normalmente no están presentes. En este ejemplo, los caracteres "menor que" (<) y ampersand (&) en el número de serie deben representarse como un triplete de escape en el URI de EPC.

7.17 Número Global de Localización de la Parte (PGLN)

El EPC PGLN (Sección 6.3.15) corresponde directamente al Número Global de Localización de una Parte (PARTE) como se especifica en las Especificaciones Generales de GS1 [GS1GS].

A continuación, se muestra gráficamente la correspondencia entre el URI de EPC PGLN y una cadena de elementos GS1 que consiste en una llave GLN de la parte (AI 417):

Figura 7-16 Correspondencia entre el URI de EPC SGLN sin extensión y la cadena de elementos GS1.



Formalmente la correspondencia se define de la siguiente manera. El URI de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de EPC: `urn:epc:id:pgln: d1d2…dL.d(L+1)d(L+2)…d12.s1s2…sK`

Cadena de elementos GS1:`(417) d1d2… d13`

Para encontrar la cadena de elementos GS1 correspondiente a un URI de EPC PGLN:

1. Numere los dígitos de los dos primeros componentes del EPC como se muestra antes. Tenga en cuenta que siempre habrá un total de 12 dígitos.
2. Calcule el dígito de verificación d_{13} = (10 - ((3(d_2 + d_4 + d_6 + d_8 + d_{10} + d_{12}) + (d_1+ d_3 + d_5 + d_7 + d_9 + d_{11})) mod 10)) mod 10.
3. Acomode los dígitos resultantes como se muestra para la cadena de elementos GS1.

Para encontrar el URI de EPC correspondiente a una cadena de elementos GS1 que incluye un GLN (AI 417):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa.
3. Acomode los dígitos como se muestra para el URI de EPC. Tenga en cuenta que el dígito de verificación del GLN d_{13} no se incluye en el URI de EPC.

Ejemplo:

URI de EPC: `urn:epc:id:pgln:1234567.89012`

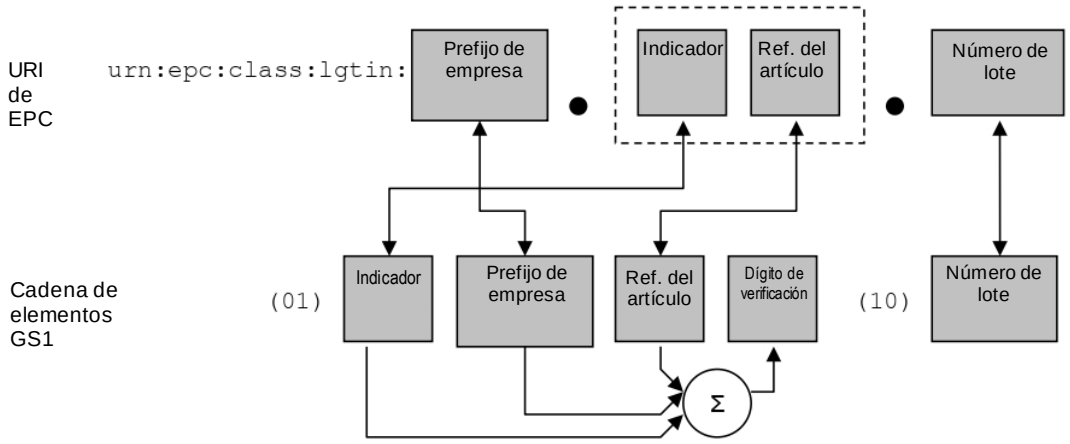
Cadena de elementos GS1: `(417) 1234567 89012 8`

7.18 GTIN + lote (LGTIN)

La clase de EPC LGTIN (Sección *6.3.1*) no corresponde directamente a ninguna llave GS1, sino que corresponde a una combinación de una llave GTIN más un Número de lote. El Número de lote en LGTIN se define como equivalente a AI 10 en las Especificaciones Generales de GS1.

A continuación, se muestra gráficamente la correspondencia entre el URI de clase de EPC LGTIN y una cadena de elementos GS1 que consiste en una llave GTIN (AI 01) y un Número de lote (AI 10):

Figura 7-17 Correspondencia entre el URI de clase de EPC LGTIN y la cadena de elementos GS1.



(Tenga en cuenta que en el caso de un GTIN-12 o GTIN-13, un carácter de relleno cero ocupa el lugar del dígito indicador en la figura anterior.)

Formalmente la correspondencia se define de la siguiente manera. El URI de clase de EPC y la cadena de elementos GS1 se deben escribir como sigue:

URI de clase de EPC: `urn:epc:class:lgtin:d2d3…d(L+1).d1d(L+2)d(L+3)…d13.s1s2…sK`

Cadena de elementos GS1: `(01)d1d2…d14 (10)s1s2…sK`

donde 1 ≤ K ≤ 20.

Para encontrar la cadena de elementos GS1 correspondiente al URI de clase de EPC LGTIN:

1. Numere los dígitos de los dos primeros componentes del URI como se muestra antes. Tenga en cuenta que siempre habrá un total de 13 dígitos.
2. Numere los caracteres del componente del Número de lote (tercero) del URI como se muestra antes. Cada *Si* corresponde a un solo carácter o a un triplete de escape porcentual que consta de un carácter % seguido de dos dígitos hexadecimales.
3. Calcule el dígito de verificación ***d***14 = (10 - ((3(***d***1 + ***d***3 + ***d***5 + ***d***7 + ***d***9 + ***d***11 + ***d***13) + (***d***2 + ***d***4 + ***d***6 + ***d***8 + ***d***10 + ***d***12)) mod 10)) mod 10.
4. Acomode los dígitos y caracteres resultantes como se muestra para la cadena de elementos GS1. Si cualquier *s* i en el URI de es un triplete de escape porcentual %xx, reemplace en la cadena de elementos GS1 el triplete con el carácter correspondiente de acuerdo con la *Tabla A-1* (para un triplete de escape porcentual %xx dado, encuentre la fila de la *Tabla A-1* que contiene xx en la columna "Valor hexadecimal"; la columna "Símbolo gráfico" proporciona el carácter correspondiente para usar en la cadena de elementos GS1.).

Para encontrar el URI de clase de EPC correspondiente a una cadena de elementos GS1 que incluye tanto un GTIN (AI 01) como un Número de lote (AI 10):

1. Numere los dígitos y caracteres de la cadena de elementos GS1 como se muestra antes.
2. Salvo en el caso de un GTIN-8, determine la cantidad de dígitos ***L*** en el Prefijo GS1 de empresa. Esto se puede realizar, por ejemplo, haciendo referencia a una tabla externa de prefijos de empresa. Consulte la Sección *7.1.2* para ver el caso de un GTIN-8.
3. Acomode los dígitos como se muestra para el URI de clase de EPC. Tenga en cuenta que el dígito de verificación del GTIN ***d***14 no se incluye en la clase URI de EPC. Reemplace cada carácter de número de serie si con el valor correspondiente en la **columna** "Forma URI" de la *Tabla A-1* , ya sea el carácter en sí o un triplete escape porcentual si si no es un carácter URI legal.

Ejemplo:

URI de clase de EPC: `urn:epc:class:lgtin:0614141.712345.32a%2Fb`

Cadena de elementos GS1: `(0-) 7 0614141 12345 1 (10) 32a/b`

Se han agregado espacios a la cadena de elementos GS1 para mayor claridad, pero normalmente no están presentes. En este ejemplo, el carácter de barra diagonal (/) del número de serie debe representarse como un triplete de escape en el URI de clase de EPC.

Para GTIN-12, GTIN-13, GTIN-8 y otras formas del GTIN, consulte las subsecciones de la Sección 7.1. Las consideraciones de esas secciones se aplican de manera análoga a LGTIN.

8 URI para patrones de identidad pura de EPC

En algunas aplicaciones de software se deben especificar reglas para filtrar listas de identidades puras EPC de acuerdo con varios criterios. Esta especificación proporciona una forma de URI de Patrón de identidad pura para este fin. Un URI de Patrón de identidad pura no representa un único EPC, sino que se refiere a un conjunto de EPC. Un URI de Patrón de identidad pura típico se ve de la siguiente manera:

```
urn:epc:idpat:sgtin:0652642.*.*
```

Este patrón se refiere a cualquier EPC SGTIN, cuyo Prefijo GS1 de empresa es 0652642 y cuya Referencia del Artículo y Número de Serie pueden ser cualesquiera. La longitud de la etiqueta y los bits de filtro no se consideran en lo absoluto al hacer coincidir el patrón con los EPC.

En general, existe un esquema de URI de Patrón de identidad pura correspondiente a cada esquema de URI de Identidad Pura EPC (Sección 6.3), cuya sintaxis es esencialmente idéntica, excepto que cualquier número de campos que comiencen a la derecha puede ser un asterisco (*). Esto es más restrictivo que los URI de Patrón de Etiquetas EPC (Sección 13), ya que los caracteres de asterisco deben ocupar los campos adyacentes más a la derecha y la sintaxis de rango no está permitida en lo absoluto.

El URI de patrón de identidad pura para la Construcción del DOD es el siguiente:

```
urn:epc:idpat:uSdod:CAGECodeOrDODAACPat.SerialNumberPat
```

con restricciones similares en cuanto al uso de asterisco (*).

8.1 Sintaxis

A continuación, se muestra la sintaxis de los URI de Patrón de identidad pura.

```
IDPatURI ::= "urn:epc:idpat:" IDPatBody

IDPatBody ::= GIDIDPatURIBody | SGTINIDPatURIBody | SGLNIDPatURIBody |
GIAIIDPatURIBody | SSCCIDPatURIBody | GRAIIDPatURIBody | GSRNIDPatURIBody
GSRNPIDPatURIBody | GDTIIDPatURIBody | SGCNIDPatURIBody | GINCIDPatURIBody
GSINIDPatURIBody | DODIDPatURIBody | ADIIDPatURIBody | CPIIDPatURIBody |
ITIPIDPartURIBody | UPUIIDPatURIBody| PGLNIDPatURIBody

GIDIDPatURIBody ::= "gid:" GIDIDPatURIMain

GIDIDPatURIMain ::=
  2*(NumericComponent ".") NumericComponent
 | 2*(NumericComponent ".") "*"
 | NumericComponent ".*.*"
 | "*.*.*"

SGTINIDPatURIBody ::= "sgtin:" SGTINPatURIMain

SGTINPatURIMain ::=
  2*(PaddedNumericComponent ".") GS3A3Component
 | 2*(PaddedNumericComponent ".") "*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

GRAIIDPatURIBody ::= "grai:" SGLNGRAIIDPatURIMain

SGLNIDPatURIBody ::= "sgln:" SGLNGRAIIDPatURIMain

SGLNGRAIIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponentOrEmpty "."
GS3A3Component
 | PaddedNumericComponent "." PaddedNumericComponentOrEmpty ".*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

SSCCIDPatURIBody ::= "sscc:" SSCCIDPatURIMain

SSCCIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponent
 | PaddedNumericComponent ".*"
 | "*.*"

GIAIIDPatURIBody ::= "giai:" GIAIIDPatURIMain

GIAIIDPatURIMain ::=
  PaddedNumericComponent "." GS3A3Component
 | PaddedNumericComponent ".*"
 | "*.*"

GSRNIDPatURIBody ::= "gsrn:" GSRNIDPatURIMain

GSRNPIDPatURIBody ::= "gsrnp:" GSRNIDPatURIMain
```

```
GSRNIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponent
 | PaddedNumericComponent ".*"
 | "*.*"

GDTIIDPatURIBody ::= "gdti:" GDTIIDPatURIMain

GDTIIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponentOrEmpty "."
GS3A3Component
 | PaddedNumericComponent "." PaddedNumericComponentOrEmpty ".*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

CPIIDPatURIBody ::= "cpi:" CPIIDPatMain

CPIIDPatMain ::=
  PaddedNumericComponent "." CPRefComponent "." NumericComponent
 | PaddedNumericComponent "." CPRefComponent ".*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

SGCNIDPatURIBody ::= "sgcn:" SGCNIDPatURIMain

SGCNIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponentOrEmpty "."
PaddedNumericComponent
 | PaddedNumericComponent "." PaddedNumericComponentOrEmpty ".*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

GINCIDPatURIBody ::= "ginc:" GINCIDPatURIMain

GINCIDPatURIMain ::=
  PaddedNumericComponent "." GS3A3Component
 | PaddedNumericComponent ".*"
 | "*.*"

GSINIDPatURIBody ::= "gsin:" GSINIDPatURIMain

GSINIDPatURIMain ::=
  PaddedNumericComponent "." PaddedNumericComponent
 | PaddedNumericComponent ".*"
 | "*.*"

ITIPIDPatURIBody ::= "itip:" ITIPPatURIMain

ITIPPatURIMain ::=
  4*(PaddedNumericComponent ".") GS3A3Component
  4*(PaddedNumericComponent ".") "*"

 | 2*(PaddedNumericComponent ".") "*.*.*"
 | PaddedNumericComponent ".*.*.*.*"
 | "*.*.*.*.*"

UPUIIDPatURIBody ::= "upui:" UPUIPatURIMain

UPUIPatURIMain ::=
  2*(PaddedNumericComponent ".") GS3A3Component
 | 2*(PaddedNumericComponent ".") "*"
 | PaddedNumericComponent ".*.*"
 | "*.*.*"

PGLNIDPatURIBody ::= "pgln:" PGLNPatURIMain

PGLNPatURIMain ::=
  2*(PaddedNumericComponent "."
 | 2*(PaddedNumericComponent ".")
```

```
  | PaddedNumericComponent ".*"
  | "*.*"

DODIDPatURIBody ::= "usdod:" DODIDPatMain

DODIDPatMain ::=
   CAGECodeOrDODAAC "." DoDSerialNumber
  | CAGECodeOrDODAAC ".*"
  | "*.*"

ADIIDPatURIBody ::= "adi:" ADIIDPatMain

ADIIDPatMain ::=
   CAGECodeOrDODAAC "." ADIComponent "." ADIExtendedComponent
  | CAGECodeOrDODAAC "." ADIComponent ".*"
  | CAGECodeOrDODAAC ".*.*"
  | "*.*.*"
```

8.2 Semántica

Formalmente, se define el significado de un URI de Patrón de identidad pura `(urn:epc:idpat:)` como que denota un conjunto de un conjunto de EPC de identidad pura, de manera respectiva.

El conjunto de EPC denotado con un URI de Patrón de identidad pura específico se define por el procedimiento de decisión siguiente, que indica si un URI de Identidad Pura de EPC corresponde al conjunto denotado por el URI de Patrón de Identidad pura.

`urn:epc:idpat:Scheme:Pi.P2...Pn` debe ser un URI de Patrón de identidad pura. urn:epc:id:*Scheme*:Ci.C2…C*n* debe ser un `URI de Identidad Pura EPC`, donde el campo `Scheme` de ambos URI es el mismo. El número de componentes *(n)* depende del valor de `Scheme`.

Primero, se considera que cualquier componente C*i* del URI de Identidad Pura EPC ***coincide*** con el componente P*i* del URI de Patrón de identidad pura correspondiente si:

- `Pi es un NumericComponent, y Ci es igual a Pi; o`
- `Pi es un PaddedNumericComponent, y Ci es igual a Pi tanto en valor numérico como en longitud o`
- `Pi es un GS3A3Component, ADIExtendedComponent, ADIComponent, o CPRefComponent y Ci es igual a Pi, carácter por carácter; o`
- `Pi es un CAGECodeOrDODAAC, y Ci es igual a Pi; o`
- `Pi es un StarComponent (y Ci es un valor nulo)`

Entonces el URI de Identidad Pura EPC es un integrante del conjunto que denota el URI del Patrón de identidad pura si y solo si `Ci` coincide con Pi para `1 ≤ i ≤ n`.

9 Organización de la memoria de las etiquetas RFID Gen 2

9.1 Tipos de datos de etiquetas

Las etiquetas RFID, en particular las etiquetas RFID Gen 2, pueden contener datos de tres tipos diferentes:

- **Datos comerciales**: Información que describe el objeto físico al que está adherida la etiqueta. Esta información incluye el Código de Producto Electrónico (EPC) que identifica de forma exclusiva el objeto físico y también puede incluir otros elementos de datos que se encuentran en la etiqueta. Esta información es sobre la que actúan las aplicaciones comerciales, por lo que estos datos se transfieren comúnmente entre el nivel de captura de datos y el nivel de la aplicación comercial en una arquitectura de implementación típica. La mayoría de los datos comerciales estandarizados en una etiqueta RFID son equivalentes a los datos comerciales que se pueden encontrar en otros soportes de datos, como los códigos de barras.
- **Información de control**: Información que utilizan las aplicaciones de captura de datos para ayudar a controlar el proceso de interacción con las etiquetas.

La Información de control incluye datos que ayudan a una aplicación de captura a filtrar etiquetas de grandes poblaciones para aumentar la eficiencia de lectura, la información de manejo especial que afecta el comportamiento de la aplicación de captura, la información que controla las características de seguridad de las etiquetas, etc. La Información de control generalmente *no* se transmite directamente a las aplicaciones comerciales, aunque puede influir en la forma en que una aplicación de captura presenta los datos comerciales al nivel de la aplicación comercial. A diferencia de los datos comerciales, la información de control no tiene equivalente en códigos de barras u otros soportes de datos.

- **Información de Fabricación de Etiqueta**: Información que describe la etiqueta en sí misma, a diferencia del objeto físico al que está adherida la etiqueta. La Información de Fabricación de Etiqueta incluye una identificación del fabricante y un código que indica el modelo de etiqueta. También puede incluir información que describe las capacidades de la etiqueta, así como un número de serie único asignado en el momento de la fabricación. Por lo general, la Información de Fabricación de Etiqueta es como la Información de control en el sentido de que la utilizan las aplicaciones de captura, pero no se transmite directamente a las aplicaciones comerciales. En algunas aplicaciones, el número de serie único que puede ser parte de la Información de Fabricación de Etiqueta se utiliza de forma adicional al EPC y, por lo tanto, actúa como Datos Comerciales. Como la Información de control, la Información de Fabricación de Etiqueta no tiene equivalente en códigos de barras ni en otros soportes de datos.

Cabe señalar que estas categorías son ligeramente subjetivas y las líneas pueden estar borrosas en determinadas aplicaciones. Sin embargo, son útiles para comprender cómo están estructurados los Estándares de Datos de Etiquetas y son una buena guía para su uso correcto y eficaz.

La siguiente tabla resume la información anterior.

Tabla 9-1 Tipos de datos en una etiqueta RFID Gen 2.

Tipo de información	Descripción	Lugar en la etiqueta Gen 2	Lugar donde se usa normalmente	Código de barras equivalente
Datos Comerciales	Describe el objeto físico al que está adherida la etiqueta.	Banco de EPC (excluyendo bits de PC y XPC y valor de filtro dentro del EPC) Banco de Memoria del Usuario	Capa de Captura de Datos y capa de Aplicación Comercial	Sí: llaves GS1, Identificadores de Aplicación (AI)
Información de control	Facilita una interacción de etiquetas eficiente	Banco reservado Banco de EPC: bits PC y XPC, y valor de filtro dentro de EPC	Capa de captura de datos	No
Información de Fabricación de Etiqueta	Describe la etiqueta en sí, a diferencia del objeto físico al que está adherida.	Banco de TID	Capa de Captura de Datos El número de serie de fabricación de etiquetas único puede llegar a la capa de Aplicación Comercial	No

9.2 Mapa de memoria de etiquetas Gen 2

Las estructuras de datos binarios definidas en la etiqueta estándar de datos están diseñadas para su uso en etiquetas RFID, particularmente en etiquetas UHF Clase 1 Gen 2 (también conocidas como etiquetas ISO 18000-6C). El estándar de interfaz aire [UHFC1G2] especifica la estructura de la memoria en las etiquetas Gen 2. Específicamente, establece que la memoria en estas etiquetas consta de cuatro bancos direccionables por separado, numerados 00, 01, 10 y 11. Asimismo, establece el uso previsto de cada banco y las restricciones sobre el contenido de cada banco dictadas por el comportamiento de la interfaz aire. Por ejemplo, el diseño y el significado del banco reservado (banco 00), que contiene contraseñas que rigen determinados comandos de la interfaz aérea, está completamente especificado en [UHFC1G2].

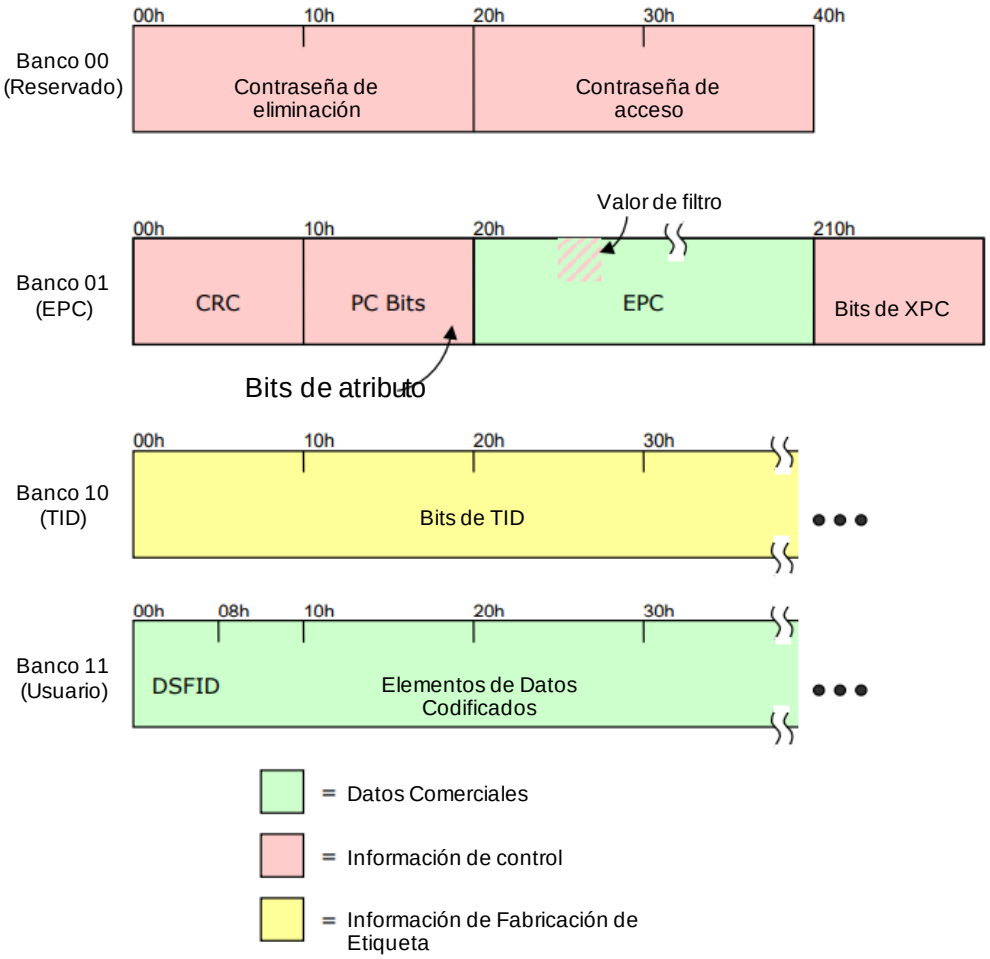
Para aquellos bancos de memoria y ubicaciones de memoria que no tienen un significado especial para la interfaz aire (es decir, son “solo datos” en lo que respecta a la interfaz aire), el Estándar de Datos de Etiquetas especifica el contenido y el significado de estas localizaciones de memoria.

Conforme a la convención establecida en [UHFC1G2], las direcciones de memoria se describen utilizando direcciones de bits hexadecimales, en donde cada banco comienza con el bit 00h y se extiende hacia arriba hasta tantos bits como contenga cada banco, con la capacidad de cada banco limitada en algunos aspectos por [UHFC1G2], pero, en última instancia, puede variar en cada marca y modelo de etiqueta. El Bit 00_h se considera el bit más significativo de cada banco, y cuando los campos binarios se colocan en la memoria de etiquetas, el bit más significativo de cualquier campo dado ocupa la dirección de bit con el número más bajo ocupada por ese campo. Sin embargo, cuando se describen campos individuales, el bit menos significativo se numera con el cero.

Por ejemplo, la contraseña de acceso es un entero sin signo de 32 bits que se compone de los bits ***b***31***b***30…***b***0, donde ***b***31 es el bit más significativo y ***b***0 es el bit menos significativo. Cuando la contraseña de acceso se almacena en la dirección 20h - 3Fh (inclusive) en el banco reservado de una etiqueta Gen 2, el bit más significativo *b*31 se almacena en la dirección de etiqueta 20h y el bit menos significativo ***b***0 se almacena en la dirección 3Fh.

El siguiente diagrama muestra el diseño de la memoria en una etiqueta Gen 2. Los colores indican el tipo de datos conforme a la categorización de la *Figura 3-1*.

Figura 9-1 Mapa de memoria de etiquetas Gen 2



La siguiente tabla describe los campos en el mapa de memoria anterior.

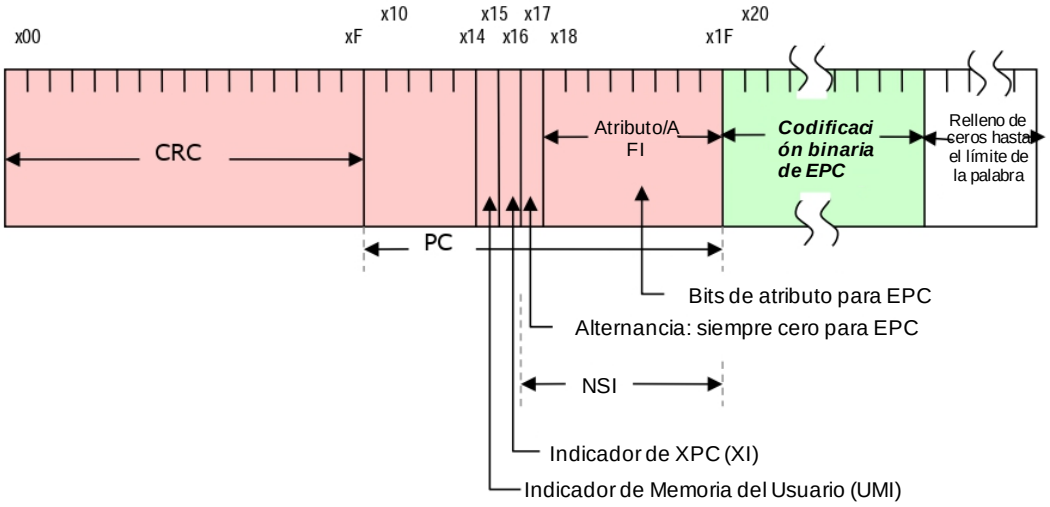
Tabla 9-2 Mapa de memoria Gen 2

Banco	Bits	Campo	Descripción	Categoría	Lugar donde se especifica
Banco 00 (Reservado)	00_h - $1F_h$	Contraseña de eliminación	Una contraseña de 32 bits que se debe presentar en la etiqueta para completar el comando de eliminación (“kill”) Gen 2.	Información de control	[UHFC1G2]
	20_h - $2F_h$	Contraseña de acceso	Una contraseña de 32 bits que se debe presentar en la etiqueta para realizar operaciones privilegiadas.	Información de control	[UHFC1G2]
Banco 01 (EPC)	00_h - $0F_h$	CRC	Una Verificación de Redundancia Cíclica de 16 bits calculada sobre el contenido del banco EPC.	Información de control	[UHFC1G2]

Banco	Bits	Campo	Descripción	Categoría	Lugar donde se especifica
	10_h - $1F_h$	Bits de PC	Bits de Control de Protocolo (véase más abajo)	Información de control	(véase más abajo)
	20_h - fin	EPC	Código de Producto Electrónico, más valor de filtro. El Código de Producto Electrónico es un identificador único global para el objeto físico al que está adherida la etiqueta. El valor del filtro proporciona un medio para mejorar la eficacia de lectura de etiquetas mediante la selección de un subconjunto de etiquetas de interés.	Datos Comerciales (excepto el valor del filtro, que es Información de control).	El EPC se define en las Secciones *6*, *7*, y *13*. Los valores de filtro se definen en la Sección *10*.
	210_h - $21F_h$	Bits de XPC	Bits de Control de Protocolo Extendido. Si el bit 16_h del banco EPC se establece en uno, los bits 210_h - $21F_h$ (inclusive) contienen bits de control de protocolo adicionales según se especifica en [UHFC1G2]	Información de control	[UHFC1G2]
Banco 10 (TID)	00_h - fin	Bits de TID	Bits de Identificación de Etiqueta, que proporcionan información sobre la etiqueta en sí, en contraposición al objeto físico al que se adhiere la etiqueta.	Información de Fabricación de Etiqueta	Sección *16*
Banco 11 (Usuario)	00_h – fin	DSFID	Lógicamente, el contenido de la memoria del usuario es un conjunto de pares de nombre-valor, donde la parte del nombre es un OID [ASN.1] y el valor es una cadena de caracteres. En términos físicos, los primeros bits son un Identificador de Formato de Almacenamiento de Datos como se especifica en [ISO15961] e [ISO15962]. El DSFID especifica el formato para el resto del banco de memoria del usuario. Normalmente tiene una longitud de ocho bits, pero puede extenderse más conforme a la especificación en [ISO15961]. Cuando el DSFID especifica el Método de Acceso 2, el formato del resto de la memoria del usuario es “Objetos Empaquetados” como se especifica en la Sección *17*. Se recomienda este formato para su uso en aplicaciones de EPC. La codificación física en el formato de datos de Objetos Empaquetados es como una secuencia de “Objetos Empacados”, donde cada uno incluye uno o más pares de nombre-valor cuyos valores se compactan.	Datos Comerciales	[ISO15961], [ISO15962], Sección *17*

El siguiente diagrama ilustra con mayor detalle los primeros bits del Banco de EPC (Banco 01) y, en particular, muestra los diversos campos dentro de los bits de Control de Protocolo (bits 10h - 1Fh, inclusive).

Figura 9-2 Mapa de memoria de Bits de Control de Protocolo (PC) Gen 2



La siguiente tabla especifica el significado de los bits de PC:

Tabla 9-3 La siguiente tabla especifica el significado de los bits de PC:

Bits	Campo	Descripción	Lugar donde se especifica
10_h – 14_h	Longitud	Representa el número de palabras de 16 bits que comprenden el campo de PC y el campo de EPC (a continuación). Consulte la Sección *15.1.1* para la ver la codificación de este campo.	[UHFC1G2]
15_h	Indicador de Memoria del Usuario (UMI)	Indica si el banco de memoria del usuario está presente y contiene datos.	[UHFC1G2]
16_h	Indicador de XPC (XI)	Indica si hay un XPC presente	[UHFC1G2]
17_h	Conmutación	Si es cero, indica una aplicación de EPCglobal; en particular, indica que los bits 18_h - $1F_h$ contienen los Bits de Atributo y el resto del banco de EPC contiene un EPC codificado en binario. Si es uno, indica una aplicación que no pertenece a EPCglobal; en particular, indica que los bits 18_h - $1F_h$ contienen el Identificador de la Familia de Aplicaciones (AFI) ISO como se define en [ISO15961] y el resto del banco EPC contiene un Identificador de Artículo Único (UII) apropiado para ese AFI.	[UHFC1G2]
18_h - $1F_h$ (si alternancia = 0)	Bits de Atributo	Bits que pueden guiar el manejo del objeto físico al que está adherida la etiqueta (se aplica solo a las etiquetas Gen2 v 1.x.).	Sección *11*
18_h - $1F_h$ (si alternancia = 1)	AFI	Un Identificador de la Familia de Aplicaciones que especifica una aplicación que no pertenece a EPCglobal para la cual el resto del banco de EPC está codificado.	[ISO15961]

Los bits 17_h - iF_h (inclusive) se conocen en conjunto como el Identificador del Sistema de Numeración (NSI). Sin embargo, debe tenerse en cuenta que cuando el bit de alternancia (bit 17_h) es cero, el sistema de numeración es siempre el Código de Producto Electrónico, y los bits 18_h - $1F_h$ contienen los Bits de Atributo cuyo propósito no está relacionado por completo con la identificación del sistema de numeración que se está utilizando.

10 Valor de filtro

El valor de filtro es información de control adicional que puede incluirse en el banco de memoria EPC de una etiqueta Gen 2. El valor de filtro tiene el objetivo de usarse para permitir que un lector de RFID seleccione o deseleccione las etiquetas correspondientes a determinados objetos físicos, a fin de facilitar la lectura de las etiquetas deseadas en un entorno donde puede haber otras etiquetas presentes en el entorno. Por ejemplo, si el objetivo es leer la etiqueta única en una tarima, y se espera que haya cientos o miles de etiquetas a nivel de artículo presentes, el rendimiento de la aplicación de captura puede mejorarse mediante el uso de la interfaz aire Gen 2 para seleccionar la etiqueta de tarima y deseleccionar las etiquetas a nivel de artículo.

Los valores de filtro están disponibles para todos los tipos de EPC excepto para el Identificador General (GID). Existe un conjunto diferente de valores normalizados para el valor de filtro asociados a cada tipo de EPC, como se especifica más abajo.

Es esencial comprender que el valor de filtro es una "información de control" adicional que ***no*** forma parte del Código Electrónico de Producto. El valor de filtro no contribuye a la identidad única del EPC. Por ejemplo, ***no*** se permite adjuntar dos etiquetas RFID a objetos físicos distintos en donde ambas etiquetas contengan el mismo EPC, incluso si los valores de filtro son diferentes en las dos etiquetas.

Debido a que el valor de filtro no forma parte del EPC, *no* se incluye cuando el EPC se representa como un URI de identidad pura, ni las aplicaciones comerciales deberían considerar el valor de filtro como parte del EPC. Sin embargo, está bien si las aplicaciones de captura leen el valor de filtro y lo elevan a las aplicaciones comerciales en algún campo de datos que no sea el EPC. Pese a esto, debe reconocerse que el objetivo de los valores de filtro es ayudar en el proceso de captura de datos y, en la mayoría de los casos, los valores de filtro serán de valor limitado o nulo para las aplicaciones comerciales. El valor de filtro ***no*** está destinado a proporcionar un indicador a nivel de empaquetado confiable para que lo utilicen las aplicaciones comerciales.

Las tablas de valores de filtro para todos los esquemas de EPC están disponibles para su descarga en http://www.gs1.org/gsmp/kc/epcglobal/tds.

10.1 Uso de valores de filtro "Reservado" y "Todos los demás"

En las siguientes secciones, los valores de filtro marcados como "reservados" están reservados para que EPCglobal los asigne en versiones futuras de esta especificación. Las implementaciones de las reglas de codificación y decodificación especificadas en este documento DEBEN aceptar cualquier valor de los valores de filtro, sea reservado o no. Sin embargo, las aplicaciones NO DEBERÍAN ordenarle a un codificador que escribiera un valor reservado en una etiqueta ni depender en un valor reservado decodificado de una etiqueta, ya que hacerlo puede causar problemas de interoperabilidad si se asigna un valor reservado en una revisión futura de esta especificación.

Cada esquema de EPC incluye un valor de filtro identificado como "Todos los demás". Este valor de filtro significa que el objeto al que está adherida la etiqueta no coincide con la descripción de ninguno de los otros valores de filtro definidos para ese esquema de EPC. En algunos casos, el valor de filtro "Todos los demás" puede aparecer en una etiqueta que se codificó para cumplir con una versión anterior de esta especificación, momento en el que no se disponía de ningún otro valor de filtro adecuado. Cuando se codifica una etiqueta nueva, debe establecerse el valor del filtro para que coincida con la descripción del objeto al que está adherida la etiqueta y solo se debe utilizar "Todos los demás" si en esta especificación no se define un valor de filtro adecuado para el objeto.

10.2 Valores de filtro para las etiquetas EPC SGTIN

A continuación, se especifican las especificaciones normativas para los valores de filtro para las etiquetas EPC SGTIN.

Tabla 10-1 Valores de filtro de SGTIN

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección 10.1)	0	000
Artículo Comercial de Punto de Venta (POS)	1	001
Caja Completa para Transporte *	2	010
Reservado (ver Sección 10.1)	3	011
Agrupación de Artículos Comerciales de Paquete Interno para su Manipulación	4	100
Reservado (ver Sección 10.1)	5	101
Carga Unitaria **	6	110

Tipo	Valor de filtro	Valor binario
Unidad dentro del Artículo Comercial o componente dentro de un producto no destinado a venta individual	7	111

* Cuando se usa como el Valor de Filtro de EPC para un SGTIN, "**Caja Completa para Transporte**" denota una caja cuya composición de múltiples artículos comerciales de POS se estandariza mediante datos maestros y puede (re)ordenarse sistemáticamente en esta configuración mediante la referencia de un solo GTIN.

** Cuando se usa como el valor de filtro EPC para un SGTIN, "**Unidad de carga**" denota uno o más artículos comerciales contenidos en una tarima u otro tipo de portador de carga (por ejemplo, plataforma rodante, carretilla de mano, contenedor, perchero, bolsa, saco, etc.) *, lo que los hace adecuados para transportarlos, apilarlos y almacenarlos como una unidad, cuya composición se estandariza a través de datos maestros y puede (re)ordenarse sistemáticamente en esta configuración mediante la referencia de un solo GTIN.

10.3 Valores de filtro para las etiquetas EPC SSCC

A continuación, se especifican las especificaciones normativas para los valores de filtro para las etiquetas EPC SSCC.

Tabla 10-2 Valores de filtro de SSCC

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Caja Completa para Transporte	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Carga Unitaria	6	110
Reservado (ver Sección *10.1*)bookmark228	7	111

10.4 Valores de filtro para las etiquetas EPC SGLN

Tabla 10-3 Valores de filtro SGLN

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)bookmark228	7	111

10.5 Valores de filtro para las etiquetas EPC GRAI

Tabla 10-4 Valores de filtro GRAI

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010

Tipo	Valor de filtro	Valor binario
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.6 Valores de filtro para las etiquetas EPC GIAI

Tabla 10-5 Valores de filtro GIAI

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Vehículo Ferroviario	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.7 Valores de filtro para las etiquetas EPC GSRN y GSRNP

Tabla 10-6 Valores de filtro GSRN y GSRNP

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.8 Valores de filtro para las etiquetas EPC GDTI

Tabla 10-7 Valores de filtro GDTI

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Documento de Viaje *	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110

Tipo	Valor de filtro	Valor binario
Reservado (ver Sección *10.1*)	7	111

* Un **Documento de viaje** es un documento de identidad emitido por un gobierno o una organización de tratado internacional para facilitar el movimiento de personas a través de fronteras internacionales.

10.9 Valores de filtro para las etiquetas EPC CPI

Tabla 10-8 Valores de filtro CPI

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.10 Valores de filtro para las etiquetas EPC SGCN

Tabla 10-9 Valores de filtro SGCN

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.11 Valores de filtro para las etiquetas EPC ITIP

Tabla 10-10 Valores de filtro ITIP

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000
Reservado (ver Sección *10.1*)	1	001
Reservado (ver Sección *10.1*)	2	010
Reservado (ver Sección *10.1*)	3	011
Reservado (ver Sección *10.1*)	4	100
Reservado (ver Sección *10.1*)	5	101
Reservado (ver Sección *10.1*)	6	110
Reservado (ver Sección *10.1*)	7	111

10.12 Valores de filtro para las etiquetas EPC GID

El esquema de EPC GID no prevé el uso de valores de filtro.

10.13 Valores de filtro para las etiquetas EPC DOD

Los valores de filtro para las etiquetas EPC del DOD de los EE. UU. son los especificados en [USDOD].

10.14 Valores de filtro para las etiquetas EPC ADI

Tabla 10-11 Valores de filtro ADI

Tipo	Valor de filtro	Valor binario
Todos los demás (ver Sección *10.1*)	0	000000
Artículo, que no sea un elemento al que se aplican los valores de filtro del 8 al 63	1	000001
Cartón	2	000010
Reservado (ver Sección *10.1*)	3 a 5	000011 a 000101
Tarima	6	000110
Reservado (ver Sección *10.1*)	7	000111
Cojines de asiento	8	001000
Cubreasientos	9	001001
Cinturones de seguridad	10	001010
Galera, carros de galera y otros equipos de servicio de galera	11	001011
Dispositivos de carga unitaria, contenedores de carga	12	001100
Artículos de seguridad de aeronave (cajas de chalecos salvavidas, paredes traseras de lavabos, trampillas de acceso al techo de los lavabos)	13	001101
Chalecos salvavidas	14	001110
Generadores de oxigeno	15	001111
Componentes del motor	16	010000
Aviónica	17	010001
Equipo experimental ("prueba de vuelo").	18	010010
Otros equipos de emergencia (máscaras de humo, PBE, hachas de choque, botiquines médicos, detectores de humo, linternas, tarjetas de seguridad, etc.).	19	010011
Otros rotativos; por ejemplo, línea o base reemplazable.	20	010100
Otros reparables	21	010101
Otro interior de cabina	22	010110
Otra reparación (excluir componente); por ejemplo, reparación de elementos de estructura	23	010111
Asientos de pasajero (estructura)	24	011000
Sistemas de IFE (entretenimiento en vuelo)	25	011001
Reservado (ver Sección *10.1*)	26 a 55	011010 a 110111
Identificador de localización (*)	56	111000
Documentación	57	111001
Herramientas	58	111010
Equipos de apoyo terrestre	59	111011
Otro equipo que no se puede volar	60	111100
Reservado para uso interno de la empresa	61 a 63	111101 a 111111

Reglas no normativas: Al asignar valores de filtro a partes etiquetadas, los valores de filtro elegidos deben ser lo más específicos posible. Por ejemplo, un valor de filtro de 17 (Aviónica) es una mejor opción para una caja negra de radar que la categoría más general de 20 (Otros rotables). Por otro lado, un valor de filtro de 20 (Otros rotables) sería adecuado para una antena de radar en el cono de nariz de un avión, ya que 17 (Aviónica) no sería exacto.

* **Nota**: el identificador de localización puede actuar de manera diferente a la etiqueta de "identificación" de un artículo en el sentido de que identifica una localización a la que otros artículos pueden hacer referencia. Por lo tanto, un artículo podría tener una etiqueta de identificación, pero también una etiqueta de localización. Un ejemplo podría ser una parte en particular de un avión o incluso el avión completo.

Reglas no normativas: Un ejemplo de "localización" podría ser el "número de cola" de un avión en particular. Por ejemplo, la Aerolínea XYZ tiene una flota de 200 aviones 737 con la misma configuración interior y, una vez dentro de ella, no puede saber en qué 737 en particular se encuentra. Esta aerolínea quiere colocar "marcadores de localización" RFID con el número de cola codificado dentro de las puertas de pasajeros o las pruebas de la bodega de carga. Las puertas podrían terminar con dos etiquetas, una para la puerta en sí misma, es decir, la que tiene el número de parte de la puerta, el número de serie, etc., y otra etiqueta que es para fines de "localización".

11 Bits de atributo

Esta sección se aplica únicamente a las etiquetas Gen2 v 1.x.

Los bits de atributo son ocho bits de "información de control" que pueden ser utilizados por aplicaciones de captura para guiar el proceso de captura. Estos bits pueden emplearse para determinar si el objeto físico al que se adhiere una etiqueta requiere un manejo especial de algún tipo.

Los bits de atributo están disponibles para todos los tipos de EPC. Se aplican las mismas definiciones de bits de atributo que se especifican a continuación, independientemente del esquema de EPC que se utilice.

Es esencial comprender que los bits de atributo son "información de control" adicional que no forma parte del Código Electrónico de Producto. Los bits de atributo no contribuyen a la identidad única del EPC. Por ejemplo, no se permite adjuntar dos etiquetas RFID a dos objetos físicos distintos en donde ambas etiquetas contengan el mismo EPC, incluso si los bits de atributo son diferentes en las dos etiquetas.

Debido a que los bits de atributo no forman parte del EPC, no se incluyen cuando el EPC se representa como un URI de identidad pura, ni las aplicaciones comerciales deberían considerar los bits de atributo como parte del EPC. Sin embargo, está bien si las aplicaciones de captura leen los bits de atributo y los elevan a las aplicaciones comerciales en algún campo de datos que no sea el EPC. Pese a esto, debe reconocerse que el objetivo de los bits de atributo es ayudar en el proceso de captura de datos y manipulación física y, en la mayoría de los casos, los bits de atributo serán de valor limitado o nulo para las aplicaciones comerciales. Los bits de atributo no están destinados a proporcionar datos maestros confiables o atributos descriptivos del producto para que los utilicen las aplicaciones comerciales.

Los bits de atributo asignados actualmente son los que se especifican a continuación:

Tabla 11-1 Asignaciones de bits de atributo

Dirección de bit	Asignado a partir de la versión TDS	Significado
18_h	[sin asignar]	
19_h	[sin asignar]	
$1A_h$	[sin asignar]	
$1B_h$	[sin asignar]	
$1C_h$	[sin asignar]	
$1D_h$	[sin asignar]	
$1E_h$	[sin asignar]	

Dirección de bit	Asignado a partir de la versión TDS	Significado
$1F_h$	1.5	El bit “1” indica que la etiqueta está adherida a un material peligroso. El bit “0” no proporciona tal indicación.

En la tabla anterior, los bits de atributo marcados como “sin asignar” están reservados para que los asigne EPCglobal en versiones futuras de esta especificación. Las implementaciones de las reglas de codificación y decodificación especificadas en este documento DEBEN aceptar cualquier valor de los bits de atributo, sea reservado o no. Sin embargo, las aplicaciones DEBERÍAN ordenar a un codificador que escriba un cero para cada bit sin asignar y NO DEBERÍAN depender del valor de un bit sin asignar decodificado a partir de una etiqueta, ya que hacerlo puede causar problemas de interoperabilidad si se asigna un valor sin asignar en una revisión futura de esta especificación.

12 URI de etiqueta EPC y URI de datos sin procesar EPC

El banco de memoria EPC de una etiqueta Gen 2 contiene un EPC codificado en binario, junto con otra información de control. Por lo general las aplicaciones no procesan datos binarios directamente. Una aplicación que desee leer el EPC puede recibir el EPC como un URI de identidad pura EPC, como se define en la Sección 6. Sin embargo, en otras situaciones, una aplicación de captura puede estar interesada en la información de control en la etiqueta, así como en el EPC. Además, una aplicación que escribe el banco de memoria EPC debe especificar los valores para la información de control que se escriben junto con el EPC. En ambas situaciones, se pueden utilizar el URI de etiqueta EPC y el URI de datos sin procesar EPC.

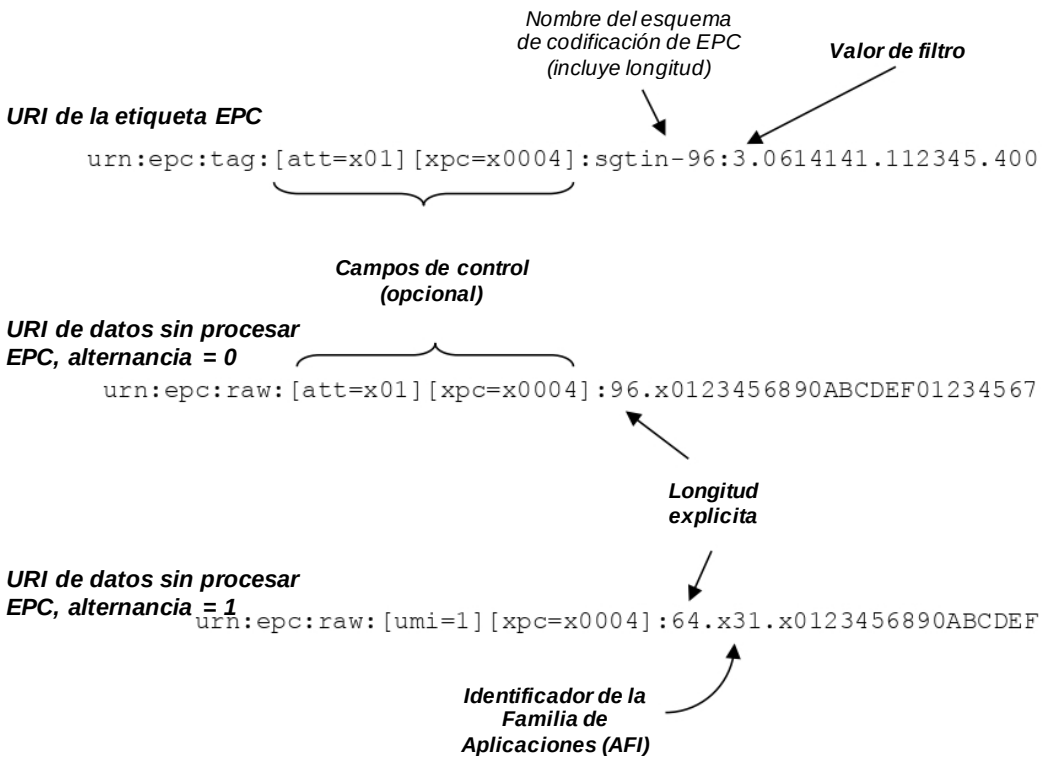
El URI de etiqueta EPC especifica tanto el EPC como los valores de la información de control en el banco de memoria EPC. También especifica cuál de las diversas variantes de esquemas de codificación binaria se utilizará (por ejemplo, la elección entre SGTIN-96 y SGTIN-198). Como tal, un URI de etiqueta EPC especifica completa y exclusivamente el contenido del banco de memoria EPC. El URI de datos sin procesar EPC también especifica el contenido completo del banco de memoria EPC, pero representa el contenido de la memoria como un solo número decimal o hexadecimal.

12.1 Estructura del URI de etiqueta EPC y el URI de datos sin procesar EPC

El URI de etiqueta EPC comienza con `urn:epc:tag:` y se utiliza cuando el banco de memoria EPC contiene un EPC válido. Los URI de etiqueta EPC se asemejan a los URI de identidad pura EPC, pero con información de control adicional. El URI de datos sin procesar EPC comienza con urn:epc:raw: y se utiliza cuando el banco de memoria EPC no contiene un EPC válido. Esto incluye situaciones en las que el bit de alternancia (bit 17h) se establece en uno, así como situaciones en las que el bit de alternancia se establece en cero, pero el resto del banco EPC no se ajusta a las reglas de codificación especificadas en la Sección 14, ya sea porque los bits de encabezado no están asignados o el resto de la codificación binaria viola una verificación de validez para ese encabezado.

La siguiente figura ilustra estas formas de URI.

Figura 12-1 Ilustración del URI de etiqueta EPC y el URI de datos sin procesar EPC



La primera forma de la figura, el URI de etiqueta EPC, se utiliza para un EPC válido. Se parece al URI de identidad pura de EPC, con la adición de campos de información de control opcionales como se especifica en la Sección 12.2.2 y un valor de filtro (no opcional). El nombre del esquema de EPC (sgtin-96 en el ejemplo anterior) especifica un esquema de codificación binario particular, por lo que incluye la longitud de la codificación. Esto contrasta con el URI identidad pura EPC que identifica un esquema de EPC pero no una codificación binaria específica (por ejemplo, `sgtin` pero no específicamente `sgtin-96`).

El URI de datos sin procesar EPC ilustrado por el segundo ejemplo en la figura puede usarse siempre que el bit de alternancia (bit 17_h) sea cero, pero generalmente solo se usa si la primera forma no puede usarse (es decir, si el contenido del banco de EPC no puede decodificarse conforme a la Sección 14.3.9). Especifica el contenido del bit 20_h en adelante como un único número hexadecimal. El número de bits en este numeral está determinado por el campo "longitud" en el banco de EPC de la etiqueta (bits 10_h - 14_h). (La sintaxis de la Sección 12.4 incluye una variante de esta forma en la que el contenido se especifica como un número decimal; esta forma es obsoleta).

El URI de datos sin procesar EPC ilustrado por el tercer ejemplo en la figura se usa cuando el bit de alternancia (bit 17_h) es uno. Es similar a la segunda forma, pero con un campo adicional entre la longitud y la carga útil que informa el valor del campo de AFI (bits 18_h - $1F_h$) como un número hexadecimal.

Cada una de estas formas se define por totalmente con los procedimientos de codificación y decodificación especificados en la Sección 14.5.12

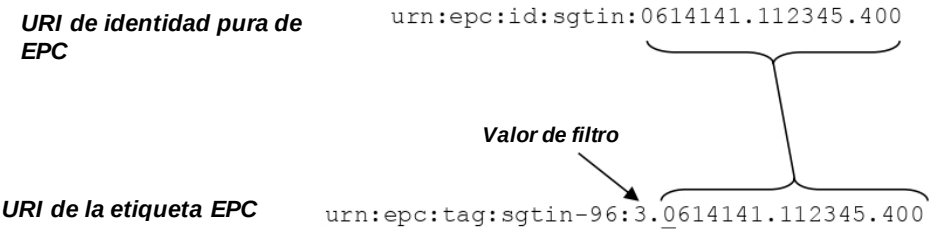
12.2 Información de control

El URI de etiqueta EPC y URI sin formato EPC especifican el contenido completo del banco de memoria EPC Gen 2, incluida la información de control, como los valores de filtro y los bits de atributo. Esta sección especifica cómo se incluye la información de control en estos URI.

12.2.1 Valores de filtro

Los valores de filtro solo están disponibles cuando el banco de EPC contiene un EPC válido y solo cuando el EPC es un esquema de EPC distinto de GID. En el URI de etiqueta EPC, el valor del filtro se indica como un campo adicional después del nombre del esquema y antes del resto del EPC, como se ilustra a continuación:

Figura 12-2 Ilustración del valor del filtro dentro del URI de etiqueta EPC



El valor del filtro es un entero decimal. Los valores permitidos del valor del filtro se especifican en la Sección *10*.

12.2.2 Otros campos de información de control

La información de control en el banco de EPC, además de los valores de filtro, se almacena por separado del EPC. Dicha información se puede representar tanto en el URI de etiqueta EPC como en el URI de datos sin procesar EPC, utilizando la sintaxis de par nombre-valor que se describe a continuación.

En ambas formas de URI, pueden aparecer pares de nombre-valor de campo de control después de urn:epc:tag: o urn:epc:raw:, tal como se muestra a continuación:

```
urn:epc:tag:[att=x0i][xpc=x0004]:sgtin-96:3.06i4i4i.ii2345.400

urn:epc:raw:[att=x0i][xpc=x0004]:96.x0i2345689ABCDEF0i234567
```

Cada elemento dentro de los corchetes especifica el valor de un campo de información de control. Un campo omitido equivale a especificar un valor igual a cero. Como caso limitante, si no se especifican campos de información de control en el URI, esto equivale a especificar un valor igual a cero para todos los campos. Esto genera compatibilidad retroactiva con las versiones preliminares del Estándar de Datos de Etiquetas.

En la tabla siguiente se especifican los campos de información de control disponibles.

Tabla 12-1 Campos de información de control

Campo	Sintaxis	Descripción	Lectura/escritura
Bits de atributo	`[att=x`*NN*`]`	Valor de los bits de atributo (bits 18_h a $1F_h$), como un numeral hexadecimal de dos dígitos *NN*. Este campo solo está disponible si el bit de oscilación (bit 17_h) es cero.	Lectura/escritura
Indicador de memoria del usuario	`[umi=`*B*`]`	Valor del bit del indicador de memoria del usuario (bit 15_h). El valor *B* es el dígito 0 o el dígito 1.	Lectura/escritura Tome en cuenta que es posible que algunas etiquetas de generación 2 ignoren el valor escrito para este bit y, en su lugar, calculen el valor del bit a partir del contenido de la memoria del usuario. Consulte [UHFC1G2].
Bits de PC extendido	`[xpc=x`*NNNN*`]`	Valor de los bits de XPC (bits 210_h a $21F_h$) como un numeral hexadecimal de cuatro dígitos *NNNN*.	Lectura solamente

La etiqueta calcula el indicador de memoria del usuario y los bits de PC extendido como una función de otra información en la etiqueta o según las operaciones realizadas en la etiqueta (como una repetición de puesta en servicio). Por lo tanto, estos campos no se pueden escribir directamente.

Al realizar una lectura a partir de una etiqueta, los campos de información de control pueden aparecer en el URI que resulte de la decodificación del banco de memoria del EPC. Al escribir una etiqueta, los campos umi y xpc son ignorados al codificar el URI en la etiqueta.

Para facilitar la decodificación, los campos de información de control que aparezcan en el URI deben aparecer en orden alfabético (en el mismo orden que la tabla anterior).

Reglas no normativas: Ejemplos: Los ejemplos siguientes ilustran el uso de los campos de información de control en el URI de etiquetas del EPC y en el URI de datos sin procesar del EPC.

```
urn:epc:tag:sgtin-96:3.06-4-4-.--2345.400
```

Esta es una etiqueta con un EPC SGTIN, bits de filtro = 3, el atributo de material peligroso se estableció en cero, sin memoria del usuario (indicador de memoria del usuario = 0), y no se repitió la puesta en servicio (PC extendido = 0). Esto ilustra la compatibilidad retroactiva con las versiones preliminares del Estándar de Datos de Etiquetas.

```
urn:epc:tag:[att=x0-]:sgtin-96:3.06-4-4-.--2345.400
```

Esta es una etiqueta con un EPC SGTIN, bits de filtro = 3, el atributo de material peligroso se estableció en uno, sin memoria del usuario (indicador de memoria del usuario = 0), y no se repitió la puesta en servicio (PC extendido = 0). Es posible que este URI haya sido especificado por una aplicación que buscaba implementar una etiqueta con el bit de material peligroso en uno y los bits de filtro y EPC que se muestran a continuación.

```
urn:epc:raw:[att=x0-][umi=-][xpc=x0004]:96.x-234567890ABCDEF0-234567
```

Esta es una etiqueta con bit de conmutación = 0, datos aleatorios en los bits 20h y posteriores (no se pueden decodificar como EPC), atributo de material peligroso establecido en uno, contenido en la memoria del usuario distinto a cero y con repetición de la puesta en servicio (según lo indica el PC extendido).

```
urn:epc:raw:[xpc=x000-]:96.xC-.x-234567890ABCDEF0-234567
```

Esta es una etiqueta con bit de conmutación = 1, indicador de familia de aplicaciones = C1 (hexadecimal) sin memoria del usuario (según lo indica el PC extendido).

12.3 URI de etiqueta de EPC y URI de identidad pura de EPC

El URI de EPC de identidad pura definido en la Sección 6 es una representación de un EPC que se utiliza en sistemas informativos. La única información en un URI de EPC de identidad pura es el EPC en sí. En contraste, el URI de etiqueta de EPC contiene información adicional: especifica el contenido de todos los campos de información de control en el banco de memoria del EPC, así como el esquema de codificación usado para codificar el EPC en sistema binario. Por lo tanto, para convertir un URI de EPC de identidad pura en un URI de etiqueta de EPC, se debe proporcionar información adicional. Por el contrario, para extraer un URI de EPC de identidad pura de un URI de etiqueta de EPC, esta información adicional se elimina. Los procedimientos en esta sección especifican cómo se llevan a cabo estas conversiones.

12.3.1 Esquemas de codificación binaria de EPC

Para cada esquema de EPC especificado en la Sección 6, hay uno o más esquemas de codificación binaria de EPC correspondientes que determinan cómo se codifica el EPC en representación binaria para usarlo en etiquetas RFID. Cuando hay más de un esquema de codificación binaria de EPC disponible para un esquema de EPC determinado, un usuario debe elegir qué esquema de codificación binaria usar. En general, los esquemas de codificación binaria más cortos dan lugar a menos bits y, por ende, permiten el uso de etiquetas RFID menos costosas que contienen menos memoria, pero que están restringidas en cuanto al rango de números de serie permitidos. Los esquemas de codificación binaria más largos permiten el rango completo de números de serie permitidos por las Especificaciones Generales de GS1, pero requieren más bits y, por ende, etiquetas RFID más costosas.

Es importante tomar en cuenta que dos EPC son iguales si los URI de EPC de identidad pura son idénticos carácter por carácter. Una codificación binaria larga (por ejemplo, SGTIN-198) ***no*** es un EPC distinto de una codificación binaria corta (por ejemplo, SGTIN-96) si el Prefijo GS1 de empresa, la referencia del artículo y los números de serie son idénticos.

En la tabla siguiente se enumeran los esquemas de codificación binaria de EPC disponibles y se indican las limitaciones impuestas en los números de serie.

Tabla 12-2 Esquemas de codificación binaria de EPC y sus limitaciones

Esquema de EPC	Esquema de codificación binaria de EPC	EPC + cómputo de bits de filtro	Incluye un valor de filtro	Limitación del número de serie
sgtin	sgtin-96	96	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{38} (es decir, el valor decimal debe ser igual o menor que 274,877,906,943).
	sgtin-198	198	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 20 caracteres alfanuméricos)
sscc	sscc-96	96	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (11 a 5 dígitos decimales incluido el dígito de extensión, según la longitud del Prefijo GS1 de empresa)
sgln	sgln-96	96	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{41} (es decir, el valor decimal debe ser igual o menor que 2,199,023,255,551).
	sgln-195	195	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 20 caracteres alfanuméricos)
grai	grai-96	96	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{38} (es decir, el valor decimal debe ser igual o menor que 274,877,906,943).
	grai-170	170	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 16 caracteres alfanuméricos)
giai	giai-96	96	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor que un límite que varía de acuerdo con la longitud del Prefijo GS1 de empresa. Ver la Sección 14.5.5.1.
	giai-202	202	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 18 a 24 caracteres alfanuméricos, según la longitud del prefijo de empresa)
gsrn	gsrn-96	96	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (11 a 5 dígitos decimales, según la longitud del Prefijo GS1 de empresa)
gsrnp	gsrnp-96	96	SÍ	Todos los valores permitidos por las Especificaciones Generales de GS1 (11 a 5 dígitos decimales, según la longitud del Prefijo GS1 de empresa)
gdti	gdti-96	96	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{41} (es decir, el valor decimal debe ser igual o menor que 2,199,023,255,551).
	gdti-113 (OBSOLETO desde TDS 1.9)	113	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 Especificaciones previas a [GS1GS12.0] (hasta 17 dígitos decimales, con o sin ceros iniciales)

Esquema de EPC	Esquema de codificación binaria de EPC	EPC + cómputo de bits de filtro	Incluye un valor de filtro	Limitación del número de serie
	gdti-i174	174	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 17 caracteres alfanuméricos)
Sgcn	Sgcn-96	96	Sí	Numérico solamente, hasta 12 dígitos decimales, con o sin ceros iniciales.
itip	itip-ii0	110	Sí	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{38} (es decir, el valor decimal debe ser igual o menor que 274,877,906,943).
	itip-2i2	212	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 20 caracteres alfanuméricos)
gid	gid-96	96	No	Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{36} (es decir, el valor decimal debe ser igual o menor que 68,719,476,735).
uSdod	uSdod-96	96	Véase la "Guía de información sobre RFID pasivo para proveedores del Departamento de Defensa de los Estados Unidos", que se puede obtener en el sitio web del Departamento de Defensa de los Estados Unidos (http://www.dodrfid.org/supplierguide.htm).	
adi	adi-var	Variable	Sí	Ver la Sección *14.5.12.1*
cpi	cpi-96	96	Sí	Número de serie: Numérico solamente, sin ceros iniciales, el valor decimal debe ser menor a 2^{31} (es decir, el valor decimal debe ser igual o menor que 2,147,483,647). La referencia de componente/pieza también se limita a valores que son numéricos solamente, sin ceros iniciales y cuya longitud es igual o menor que 15 menos la longitud del Prefijo GS1 de empresa
	cpi-var	Variable	Sí	Todos los valores permitidos por las Especificaciones Generales de GS1 (hasta 12 dígitos decimales, sin ceros iniciales).

Reglas no normativas: Explicación: Para los esquemas de EPC SGTIN, SGLN, GRAI y GIAI, de acuerdo con las Especificaciones Generales de GS1, el número de serie es una cadena alfanumérica de longitud variable. Esto significa que los números de serie *34*, *034*, *0034*, etc., son diferentes, como lo son *P34*, *34P*, *0P34*, *P034* y así sucesivamente. Para generar hasta 20 caracteres alfanuméricos, se requieren 140 bits para codificar el número de serie. Este es el motivo por el que las codificaciones binarias "largas" tienen una cantidad grande de bits. Consideraciones similares se aplicaron al esquema de EPC GDTI, salvo que el GDTI solo permite caracteres de dígitos (pero aún permite los ceros iniciales).

Para aceptar las etiquetas RFID de 96 bits muy comunes, se introdujeron esquemas de codificación binaria adicionales que solo requieren 96 bits. Para ajustar los números de serie a 96 bits, algunos han sido excluidos. Las codificaciones de 96 bits de SGTIN, SGLN, GRAI, GIAI y GDTI se limitan a los números de serie que constan únicamente de dígitos, que no tienen ceros iniciales (a menos que el número de serie contenga un solo dígito *0*), y cuyo valor cuando se trata de un numeral decimal es menor que 2^B, donde B es la cantidad de bits disponibles en el esquema de codificación binaria. La opción de excluir los números de serie con ceros iniciales fue una opción de diseño arbitraria en el momento en que se codificaron las codificaciones de 96 bits por primera vez, una alternativa hubiera sido permitir los ceros iniciales, a expensas de excluir otros números de serie. Sin embargo, es imposible ignorar el hecho de que en los bits B, no hubo más de 2^B números de serie distintos.

Al codificar una codificación binaria "larga", no se pueden eliminar los ceros iniciales cuando la codificación binaria incluye caracteres de cero inicial. De igual manera, al codificar un EPC en una forma "corta" o "larga", no se pueden eliminar los ceros iniciales antes de la codificación.

Esto significa que los EPC cuyo número de serie tiene ceros iniciales solo se pueden codificar en la forma "larga".

En algunas aplicaciones, es deseable que el número de serie siempre contenga un número específico de caracteres. Los motivos de esto pueden incluir querer una longitud predecible para la cadena de URI de EPC o un tamaño predecible para un código de barras que codifique el mismo identificador. En algunas aplicaciones de código de barras, se logra usando ceros iniciales. Sin embargo, si se usan etiquetas de 96 bits, la opción de usar ceros iniciales no existe.

Por lo tanto, en aplicaciones en las que se requieren etiquetas de 96 bits y que el número de serie sea una cantidad fija de caracteres, se recomienda que se utilicen números de serie numéricos que estén en el rango de $10^D \leq serial < 10^{D+1}$, donde D es la cantidad deseada de dígitos. Por ejemplo, si se desea utilizar números de serie de 11 dígitos, una aplicación puede usar números de serie en el rango de 10,000,000,000 a 99,999,999,999. Esas aplicaciones deben usar números de serie que se ajusten a las limitaciones de las etiquetas de 96 bits. Por ejemplo, si se desea utilizar números de serie de 12 dígitos para las codificaciones SGTIN-96, los números de serie deben estar en el rango de 100,000,000,000 a 274,877,906,943.

Sin embargo, se debe recordar que muchas aplicaciones no necesitan una cantidad fija de caracteres en el número de serie, por lo que todos los números de serie del 0 al valor máximo (sin ceros iniciales) se pueden utilizar con etiquetas de 96 bits.

12.3.2 URI de identidad pura de EPC a URI de la etiqueta de EPC

Datos:

- Un UTI de identidad pura de EPC, según se especifica en la Sección 6.3. Esta es una cadena que coincide con la producción del EPC-URI de la sintaxis en la Sección 6.3.
- Una selección del esquema de codificación binaria que se usará. Este es uno de los esquemas de codificación binaria especificados en la columna "Esquema de codificación binaria de EPC" de la Tabla 12-2. El esquema de codificación binaria elegido debe ser aquel que corresponda al esquema de EPC en el URI de identidad pura de EPC.
- Un valor de filtro, si la columna "Incluye un valor de filtro" de la Tabla 12-2 indica que la codificación binaria incluye un valor de filtro.
- El valor de los bits de atributo.
- El valor del indicador de memoria del usuario.

Validación:

- La parte del número de serie del EPC (los caracteres que aparecen al lado derecho del punto) debe cumplir con las restricciones del esquema de codificación binaria seleccionado, según se especificó en la columna de "Limitación del número de serie" de la Tabla 12-2.
- El valor de filtro debe estar en el rango de 0 < *filtro* < 7.

Procedimiento:

1. Comenzando con el URI de identidad pura de EPC, reemplace el prefijo `urn:epc:id:` con `urn:epc:tag:`.
2. Reemplace el nombre del esquema de EPC con el nombre del esquema de codificación binaria de EPC seleccionado. Por ejemplo, reemplace `sgtin` con `sgtin-96` o `sgtin-198`.
3. Si el esquema de codificación binaria seleccionado incluye un valor de filtro, inserte el valor de filtro como un dígito decimal seguido del carácter de dos puntos (":") del URI, seguido de un carácter de punto (".").
4. Si los bits de atributo no son ceros, genere una cadena `[att=xNN]`, donde *NN* es el valor de los bits de atributo como un numeral hexadecimal de 2 dígitos.
5. Si el indicador de memoria del usuario no es un cero, genere una cadena `[umi=i]`.
6. Si el paso 4 o el paso 5 dan lugar a una cadena no vacía, inserte esas cadenas después del carácter de dos puntos (":") del URI, seguido de un carácter de dos puntos adicional.

7. La cadena resultante es el URI de etiqueta del EPC.

12.3.3 URI de etiqueta de EPC a URI de identidad pura de EPC

Datos:

1. Un URI de etiqueta de EPC, según se especifica en la sección 12. Esta es una cadena que coincide con la producción de TagURI de la gramática en la Sección 12.4.

Procedimiento:

1. Comenzando con el URI de etiqueta del EPC, reemplace el prefijo `urn:epc:tag:` con `urn:epc:id:`.
2. Reemplace el nombre del esquema de codificación binaria de EPC con el nombre del esquema de EPC correspondiente. Por ejemplo, reemplace `sgtin-96` o `sgtin-198` con `sgtin`.
3. Si el esquema de codificación incluye un valor de filtro, elimine el valor de filtro (el dígito que aparece después del carácter de dos puntos) y el carácter de punto (".").
4. Si el URI contiene uno o más campos de control según se especifica en la Sección 12.2.2, elimínelos junto con el carácter de dos puntos siguiente.
5. La cadena resultante es el URI de EPC de identidad pura.

12.4 Sintaxis

La sintaxis siguiente especifica la sintaxis del URI de etiqueta de EPC y el URI de datos sin procesar de EPC. La sintaxis hace referencia a los elementos gramaticales definidos en las secciones 5 y 6.3.

```
TagOrRawURI ::= TagURI | RawURI
TagURI ::= "urn:epc:tag:" TagURIControlBody
TagURIControlBody ::= ( ControlField+ ":" )? TagURIBody
TagURIBody ::= SGTINTagURIBody | SSCCTagURIBody | SGLNTagURIBody |
GRAITagURIBody | GIAITagURIBody | GDTITagURIBody | GSRNTagURIBody |
GSRNPTagURIBody | ITIPTagURIBody | GIDTagURIBody | SGCNTagURIBody |
DODTagURIBody | ADITagUriBody | CPITagURIBody
SGTINTagURIBody ::= SGTINEncName ":" NumericComponent "." SGTINURIBody
SGTINEncName ::= "sgtin-96" | "sgtin-198"
SSCCTagURIBody ::= SSCCEncName ":" NumericComponent "." SSCCURIBody
SSCCEncName ::= "sscc-96"
SGLNTagURIBody ::= SGLNEncName ":" NumericComponent "." SGLNURIBody
SGLNEncName ::= "sgln-96" | "sgln-195"
GRAITagURIBody ::= GRAIEncName ":" NumericComponent "." GRAIURIBody
GRAIEncName ::= "grai-96" | "grai-170"
GIAITagURIBody ::= GIAIEncName ":" NumericComponent "." GIAIURIBody
GIAIEncName ::= "giai-96" | "giai-202"
GSRNTagURIBody ::= GSRNEncName ":" NumericComponent "." GSRNURIBody
GSRNEncName ::= "gsrn- 96"
GSRNPEncName ::= "gsrnp-96"
GDTITagURIBody ::= GDTIEncName ":" NumericComponent "." GDTIURIBody
GDTIEncName ::= "gdti-96" | "gdti-113" | "gdti-174"
CPITagURIBody ::= CPIEncName ":" NumericComponent "." CPIURIBody
CPIEncName ::= "cpi-96" | "cpi-var"
SGCNTagURIBody ::= SGCNEncName ":" NumericComponent "." SGCNURIBody
SGCNEncName ::= "sgcn-96"
ITIPTagURIBody ::= ITIPEncName ":" NumericComponent "." ITIPURIBody
ITIPEncName ::= "itip-110" | "itip-212"
GIDTagURIBody ::= GIDEncName ":" GIDURIBody
```

```
GIDEncName ::= "gid-96"
DODTagURIBody ::= DODEncName ":" NumericComponent "." DODURIBody
DODEncName ::= "usdod-96"
ADITagURIBody ::= ADIEncName ":" NumericComponent "." ADIURIBody
ADIEncName ::= "adi-var"
RawURI ::= "urn:epc:raw:" RawURIControlBody
RawURIControlBody ::= ( ControlField+ ":")? RawURIBody
RawURIBody ::= DecimalRawURIBody | HexRawURIBody | AFIRawURIBody
DecimalRawURIBody ::= NonZeroComponent "." NumericComponent
HexRawURIBody ::= NonZeroComponent ".x" HexComponentOrEmpty
AFIRawURIBody ::= NonZeroComponent ".x" HexComponent ".x"
HexComponentOrEmpty
ControlField ::= "[" ControlName "=" ControlValue "]"
ControlName ::= "att" | "umi" | "xpc"
ControlValue ::= BinaryControlValue | HexControlValue
BinaryControlValue ::= "0" | "1"
HexControlValue ::= "x" HexComponent
```

13 URI para los patrones de codificación de etiquetas de EPC

En algunas aplicaciones de software se deben especificar reglas para filtrar listas de etiquetas de acuerdo con varios criterios. Esta especificación proporciona un URI de patrón de etiqueta de EPC para cumplir este fin. Un URI de patrón de etiqueta de EPC no representa la codificación de una sola etiqueta, sino que se refiere a un conjunto de codificaciones de etiqueta. Un patrón típico tiene el aspecto siguiente:

```
urn:epc:pat:sgtin-96:3.0652642.[102400-204700].*
```

Este patrón se refiere a cualquier etiqueta que contenga una codificación binaria de EPC SGTIN de 96 bits, cuyo campo de filtro es 3, cuyo Prefijo GS1 de empresa es 0652642, cuya referencia de artículo está en el rango de 102400 ≤ ***referencia de artículo*** ≤ 204700 y cuyo número de serie puede ser cualquiera.

En general, hay un esquema de URI de patrón de etiqueta de EPC que corresponde a cada esquema de codificación binaria de EPC, cuya sintaxis es prácticamente idéntica salvo porque se pueden usar rangos o el carácter de asterisco (*) en cada campo.

Para los patrones de SGTIN, SSCC, SGLN, GRAI, GIAI, GSRN, GDTI, SGCN e ITIP, la sintaxis del patrón restringe la manera en que se pueden combinar los comodines y los rangos. Solo se permiten dos posibilidades para el campo CompanyPrefix. Uno, puede ser un asterisco (*), en cuyo caso el campo siguiente (ItemReference, SerialReference, LocationReference,

```
AssetType, IndividualAssetReference, ServiceReference, DocumentType,
CouponReference, Piece o Total) también debe ser un asterisco. Dos, puede ser
un prefijo de empresa específico, en cuyo caso el campo siguiente puede ser un
número, un rango o una estrella. Está permitido no especificar ningún rango
para CompanyPrefix.
```

Reglas no normativas: Explicación: Debido a que el prefijo de empresa es de longitud variable, no se puede especificar un rango, ya que es podría tener longitudes diferentes. Sin embargo, cuando se especifica un prefijo de empresa en particular, es posible hacer coincidir rangos o todos los valores del campo siguiente, debido a que su longitud es fija para un prefijo de empresa determinado. El otro caso que está permitido ocurre cuando ambos campos son un asterisco, lo que funciona para todas las codificaciones de etiqueta porque los campos de etiqueta correspondientes (lo que incluye el campo Partición, cuando está presente) son ignorados.

El URI del patrón del constructo DoD es el siguiente:

```
urn:epc:pat:usdod-96:filterPat.CAGECodeOrDODAACPat.serialNumberPat
```

Donde *filterPat* es un valor de filtro, un rango de la forma [*lo-hi*] o un carácter *; `CAGECodeOrDODAACPat` es un código CAGE/DODAAC o un carácter de *; y `serialNumberPat` es un número de serie, un rango de la forma `[lo-hi]` o un carácter de *.

El URI de patrón para el Identificador Aeroespacial y de Defensa (ADI) es el siguiente:

`urn:epc:pat:adi-var:filterPat. CAGECodeOrDODAACPat. partNumberPat. SerialNumberPat`

donde *filterPat* n valor de filtro, un rango de la forma *[lo-hi]* o un carácter *; *CAGECodeOrDODAACPat* es un código CAGE/DODAAC o un carácter *; *partNumberPat* es una cadena vacía, un número de parte o un carácter *; y *SerialNumberPat* es un número de serie o un carácter *.

El URI de patrón del Identificador de componente/pieza (CPI) es el siguiente:

`urn:epc:pat:cpi-96:filterPat. CPI96PatBody.SerialNumberPat`

o

`urn:epc:pat:cpi-var: filterPat. CPIVarPatBody`

donde *filterPat* es un valor de filtro, un rango de la forma *[lo-hi]* o un carácter *; CPI96PatBody es *.* o un Prefijo GS1 de empresa seguido de un punto y un componente numérico/número de parte, un rango en la forma *[lo-hi]* o un carácter *; SerialNumberPat es un número de serie o un carácter * o un rango en la forma [*lo-hi*]; y *CPIVarPatBody* es *.*.* o un Prefijo GS1 de empresa seguido de un punto y de una referencia de componente/pieza seguido de un punto y de un número de serie de componente/pieza, un rango en la forma *[lo-hi]* o un carácter *.

13.1 Sintaxis

La sintaxis de los URI de patrón de etiqueta de EPC se define por la sintaxis que se describe a continuación.

```
PatURI ::= "urn:epc:pat:" PatBody
PatBody ::= GIDPatURIBody | SGTINPatURIBody | SGTINAlphaPatURIBody |
SGLNGRAI96PatURIBody | SGLNGRAIAlphaPatURIBody | SSCCPatURIBody |
GIAI96PatURIBody | GIAIAlphaPatURIBody | GSRNPatURIBody | GSRNPPatURIBody
|GDTIPatURIBody | CPIVarPatURIBody | SGCNPatURIBody | ITIPPatURIBody  |
USDOD96PatURIBody ITIP212PatURIBody | ADIVarPatURIBody |CPI96PatURIBody |
GIDPatURIBody ::= "gid-96:" 2*(PatComponent ".") PatComponent
SGTIN96PatURIBody ::= "sgtin-96:" PatComponent "." GS1PatBody "."
PatComponent
SGTINAlphaPatURIBody ::= "sgtin-198:" PatComponent "." GS1PatBody "."
GS3A3PatComponent
SGLNGRAI96PatURIBody ::= SGLNGRAI96TagEncName ":" PatComponent "."
GS1EPatBody "." PatComponent
SGLNGRAI96TagEncName ::= "sgln-96" | "grai-96"
SGLNGRAIAlphaPatURIBody ::= SGLNGRAIAlphaTagEncName ":" PatComponent "."
GS1EPatBody "." GS3A3PatComponent
SGLNGRAIAlphaTagEncName ::= "sgln-195" | "grai-170"
SSCCPatURIBody ::= "sscc-96:" PatComponent "." GS1PatBody
GIAI96PatURIBody ::= "giai-96:" PatComponent "." GS1PatBody
GIAIAlphaPatURIBody ::= "giai-202:" PatComponent "." GS1GS3A3PatBody
GSRNPatURIBody ::= "gsrn- 96:" PatComponent "." GS1PatBody
GSRNPPatURIBody ::= "gsrnp-96:" PatComponent "." GS1PatBody
GDTIPatURIBody ::= GDTI96PatURIBody | GDTI113PatURIBody| GDTI174PatURIBody
GDTI96PatURIBody ::= "gdti-96:" PatComponent "." GS1EPatBody "."
PatComponent
GDTI113PatURIBody ::= "gdti-113:" PatComponent "." GS1EPatBody "."
PaddedNumericOrStarComponent
GDTI174PatURIBody ::= "gdti-174:" PatComponent "." GS1EPatBody "."
GS1GS3A3PatBody
CPI96PatURIBody ::= "cpi-96:" PatComponent "." GS1PatBody "." PatComponent
```

```
CPIVarPatURIBody ::= "cpi-var:" PatComponent "." CPIVarPatBody
CPIVarPatBody ::= "*.*.*"
 | PaddedNumericComponent "." CPRefComponent "." PatComponent
SGCNPatURIBody ::= SGCN96PatURIBody
SGCN96PatURIBody ::= "sgcn-96:" PatComponent "." GS1EPatBody "."
PaddedNumericOrStarComponent
USDOD96PatURIBody ::= "usdod-96:" PatComponent "." CAGECodeOrDODAACPat "."
PatComponent
ADIVarPatURIBody ::= "adi-var:" PatComponent "." CAGECodeOrDODAACPat "."
ADIPatComponent "." ADIExtendedPatComponent
PaddedNumericOrStarComponent ::= PaddedNumericComponent
                | StarComponent
GS1PatBody ::= "*.*" | ( PaddedNumericComponent "." PaddedPatComponent )
GS1EPatBody ::= "*.*" | ( PaddedNumericComponent "."
PaddedOrEmptyPatComponent )
GS1GS3A3PatBody ::= "*.*" | ( PaddedNumericComponent "." GS3A3PatComponent )
PatComponent ::= NumericComponent
        | StarComponent
        | RangeComponent
PaddedPatComponent ::= PaddedNumericComponent
           | StarComponent
           | RangeComponent
PaddedOrEmptyPatComponent ::= PaddedNumericComponentOrEmpty
              | StarComponent
              | RangeComponent
GS3A3PatComponent ::= GS3A3Component | StarComponent
CAGECodeOrDODAACPat ::= CAGECodeOrDODAAC | StarComponent
ADIPatComponent::= ADIComponent | StarComponent
ADIExtendedPatComponent ::= ADIExtendedComponent | StarComponent
StarComponent ::= "*"
RangeComponent ::= "[" NumericComponent "-"
            NumericComponent "]"
```

Para que un `RangeComponent` sea legal, el valor numérico del primer `NumericComponent` debe ser menor o igual que el valor numérico del segundo `NumericComponent`.

13.2 Semántica

El significado de un URI de etiqueta de EPC (`urn:epc:pat:`) se define formalmente como un conjunto de URI de etiqueta de EPC.

El conjunto de EPC denotado con un URI de patrón de etiqueta de EPC específico se define por el procedimiento de decisión siguiente, que indica si un URI de etiqueta de EPC corresponde al conjunto denotado por el URI del patrón de la etiqueta de EPC.

Permita que `urn:epc:pat: EncName:P-.P2...Pn` sea un URI de patrón de etiqueta de EPC. Permita que urn:epc:tag: `EncName:C-.C2...Cn` sea un URI de etiqueta de EPC, donde el campo *`EncName`* de ambos URI es el mismo. La cantidad de los componentes (*n*) depende del valor de *`EncName`*.

Primero, se considera que cualquier componente Ci ***coincide*** con el componente P*i* del URI del patrón de etiqueta de EPC correspondiente si:

- `Pi es un NumericComponent y Ci equivale a Pi; o`
- `Pi es un PaddedNumericComponent y Ci equivale a Pi en valor numérico y en longitud; o`
- `Pi es un GS3A3Component, ADIExtendedComponent, ADIComponent o CPRefComponent y Ci equivale a Pi, carácter por carácter; o`

- *Pi* es un CAGECodeOrDODAAC, y Ci es igual a Pi; o
- *Pi* es un RangeComponent [lo-hi], y lo ≤ C*i* ≤ hi; o
- *Pi* es un StarComponent (y C*i* es un valor nulo)

Entonces el URI de etiqueta de EPC es un integrante del conjunto que denota el URI del patrón de EPC si y solo si C*i* coincide con P*i* para 1 ≤ *i* ≤ *n*.

14 Codificación binaria de EPC

En esta sección se especifica cómo se codifican los URI de etiqueta de EPC en cadenas binarias y, por el contrario, cómo se codifica una cadena binaria en un URI de etiqueta de EPC (si es posible). Las cadenas binarias definidas por los procedimientos de codificación y decodificación son ideales para usarse en el banco de memoria de EPC de una etiqueta Gen 2, según se especifica en la Sección *14.5.12*.

El procedimiento completo de codificación de un URI de etiqueta de EPC en el contenido binario del banco de memoria de EPC de una etiqueta Gen 2 se especifica en la Sección *15.1.1*. En el procedimiento de la Sección *15.1.1* se utiliza el procedimiento definido abajo en la Sección *14.3* para realizar la mayor parte del trabajo. Por el contrario, el procedimiento completo para decodificar el contenido binario del banco de memoria de EPC de una etiqueta Gen 2 en un URI de etiqueta de EPC (o URI de datos sin procesar de EPC, si es necesario) se especifica en la Sección *15.2.2*. En el procedimiento en la Sección *15.2.2* se utiliza el procedimiento definido abajo en la Sección *14.3.9* para realizar la mayor parte del trabajo.

14.1 Descripción general de la codificación binaria

La estructura general de una codificación binaria de EPC como se utiliza en una etiqueta es similar a una cadena de bits (es decir, una representación binaria), que consta de un encabezado de longitud fija seguido de una serie de campos cuya longitud general, estructura y función se determinan por el valor del encabezado. Los valores del encabezado asignado se especifican en la Sección *14.2*.

Los procedimientos para realizar conversiones entre el URI de etiqueta de EPC y la codificación binaria se especifican en la Sección *14.3* (codificación de URI a valores binarios) y en la Sección *14.3.9* (decodificación de valores binarios en URI). Los procedimientos de codificación y decodificación se generan por las tablas de codificación que se especifican en la Sección *14.4.9*. En cada tabla de codificación se especifica la estructura de los campos que prosiguen al encabezado de un valor de encabezado determinado.

Para convertir un URI de etiqueta de EPC en codificación binaria de EPC, siga el procedimiento especificado en la Sección *14.3*, que se sintetiza a continuación. Primero, la tabla de codificación adecuada se selecciona entre las tablas estipuladas en la Sección *14.4.9*. La tabla de codificación correcta es aquella en la que la entrada "Plantilla URI" coincide con el URI de etiqueta de EPC determinada. Cada columna en la tabla de codificación corresponde a un campo de bit en la codificación binaria final. En cada columna, se especifica un "método de codificación" que indica cómo calcular los bits correspondientes de la codificación binaria, cuando se introduce una parte del URI. Los detalles de la codificación para cada "método de codificación" se proporcionan en subsecciones de la Sección *14.3*.

Para convertir la codificación binaria de EPC en un URI de etiqueta de EPC, siga el procedimiento especificado en la Sección *14.3.9*, que se sintetiza a continuación. Primero, los ocho bits más importantes se buscan en la tabla de encabezados binarios de EPC (*Tabla 14-1* en la Sección *14.2*). Esto identifica el esquema de codificación de EPC, que a su vez selecciona una tabla de codificación de entre las especificadas en la Sección *14.4.9*. Cada columna en la tabla de codificación corresponde a un campo de bit en la codificación binaria de entrada. En cada columna, se especifica un "método de codificación" que indica cómo calcular una parte correspondiente del URI de salida, cuando se ingresa ese campo de bit. Los detalles de la decodificación para cada "método de codificación" se proporcionan en subsecciones de la Sección *14.3.9*.

14.2 Encabezados binarios de EPC

La estructura general de una codificación binaria de EPC como se utiliza en una etiqueta es similar a una cadena de bits (es decir, una representación binaria), que consta de un encabezado de 8 bits de longitud fija seguido de una serie de campos cuya longitud general, estructura y función se determinan por el valor del encabezado. Para fines de expansión futura, se define un valor de encabezado de 11111111, a fin de indicar que se utiliza un encabezado más largo de más de 8 bits; esto permite una expansión futura de modo que se puedan usar más de 256 valores de encabezado con encabezados más largos. Por lo tanto, la especificación actual permite hasta 255 encabezados de 8 bits, más un número no determinado actualmente de encabezados más largos.

Reglas no normativas: Nota sobre la compatibilidad retroactiva: En una versión anterior del Estándar de Datos de Etiquetas, el encabezado tenía una longitud variable, con un enfoque de niveles en el que un valor de cero en cada nivel indicaba que el encabezado se extraía del siguiente nivel más largo. Para las codificaciones definidas en la especificación previa, los encabezados eran de 2 bits o de 8 bits. Dado que el valor de cero se reserva para indicar un encabezado en el siguiente nivel más largo, el encabezado de 2 bits tuvo 3 valores posibles (01, 10 y 11, no 00), y el encabezado de 8 bits tuvo 63 valores posibles (asumiendo que los primeros 2 bits deben ser iguales a 00 y 00000000 está reservado para permitir encabezados de más de 8 bits de longitud). Los encabezados de 2 bits se usaron únicamente en conjunto con determinados encabezados binarios de EPC de 64 bits.

En esta versión del Estándar de Datos de Etiquetas, se abandonó el enfoque de encabezado por niveles. Asimismo, se volvieron obsoletas todas las codificaciones de 64 bits (incluidas todas las codificaciones con encabezados de 2 bits) y no se deben usar en aplicaciones nuevas. Para facilitar una transición ordenada, las partes del espacio del encabezado ocupadas formalmente por las codificaciones de 64 bits se reservan en esta versión del Estándar de Datos de Etiquetas, con la intención de que se puedan reclamar después de que caduque una “fecha final”. Después de la "fecha final”, las etiquetas que contengan EPC de 64 bits con encabezados de 2 bits y etiquetas con encabezados de 64 bits que comiencen con 00001 dejarán de interpretarse adecuadamente.

Los esquemas de codificación definidos en esta versión del Estándar de Datos de Etiquetas de EPC se muestran en la *Tabla 14-1*. La tabla también indica los valores de encabezado sin asignar actualmente que están "Reservados para uso futuro” (RFU). Todos los valores de encabezado que se han reservado para codificaciones anteriores de 64 bits, definidos en versiones previas del Estándar de Datos de Etiquetas de EPC, fueron finales, con fecha de entrada en vigor de 1.° de julio de 2009, según lo anunció anteriormente EPCglobal el 1.° de julio de 2006.

Tabla 14-1 Valores binarios de encabezado de EPC

Valor del encabezado (binario)	Valor del encabezado (hexadecimal)	Longitud de codificación (bits)	Esquema de codificación
0000 0000	00	NA	Etiqueta no programada
	01	NA	Reservado para uso futuro
0000 0001	02,03	NA	Reservado para uso futuro
0000 001x	04,05	NA	Reservado para uso futuro
0000 01xx	06,07	NA	Reservado para uso futuro
0000 1000	08		Reservado para uso futuro
0000 1001	09		Reservado para uso futuro
0000 1010	0A		Reservado para uso futuro
0000 1011	0B		Reservado para uso futuro
0000 1100 para 0000 1111	0C para 0F		Reservado para uso futuro
0001 0000 para 0010 1011	10 para 2B	NA NA	Reservado para uso futuro
0010 1100	2C	96	GDTI-96
0010 1101	2D	96	GSRN-96
0010 1110	2E	96	GSRNP
00101111	2F	96	USDoD-96
00110000	30	96	SGTIN-96
00110001	31	96	SSCC-96

Valor del encabezado (binario)	Valor del encabezado (hexadecimal)	Longitud de codificación (bits)	Esquema de codificación
00110010	32	96	SGLN-96
00110011	33	96	GRAI-96
00110100	34	96	GIAI-96
00110101	35	96	GID-96
00110110	36	198	SGTIN-198
0011 0111	37	170	GRAI-170
00111000	38	202	GIAI-202
00111001	39	195	SGLN-195
0011 1010	3A	113	GDTI-113 (OBSOLETO desde TDS 1.9)
00111011	3B	Variable	ADI-var
00111100	3C	96	CPI-96
00111101	3D	Variable	CPI-var
00111110	3E	174	GDTI-174
00111111	3F	96	SGCN-96
0100 0000	40	110	ITIP-110
0100 0001	41	212	ITIP-212
0100 0010 para 0111 1111	42 para 7F		Reservado para uso futuro
1000 0000 para 1011 1111	80 para BF		Reservado para uso futuro
1100 0000 para 1100 1101	C0 para CD		Reservado para uso futuro
1100 1110	CE		Reservado para uso futuro
1100 1111 para 1110 0001	CF para E1		Reservado para uso futuro
1110 0010	E2		E2 sigue estando RESERVADO PERMANENTEMENTE (PERMANENTLY RESERVED) para evitar confusiones con los primeros ocho bits de la memoria TID (Sección 16).
1110 0011 para 1111 1110	E3 para FE		Reservado para uso futuro
1111 1111	FF	NA	Reservado para uso futuro (Reservado expresamente para encabezados de más de 8 bits de longitud)

14.3 Procedimiento de codificación

El procedimiento siguiente codifica un URI de etiqueta de EPC en una cadena de bits que contiene el EPC codificado y (para los esquemas de EPC que tienen un valor de filtro) el valor de filtro. Esta cadena de bits es adecuada para guardar en el banco de memoria de EPC de una etiqueta Gen 2 que comienza en el bit 20.

Vea la Sección 15.1.1 para conocer el procedimiento completo de codificación del banco de memoria de EPC completo, que incluye la información de control que reside fuera del EPC codificado. (El procedimiento en la Sección 15.1.1 utiliza el procedimiento siguiente como subrutina.)

Datos:

- Un URI de etiqueta de EPC de la forma `urn:epc:tag:scheme:remainder`

Resultado:

- Una cadena de bits que contiene la codificación binaria de EPC del URI de etiqueta de EPC, que contiene el EPC codificado junto con el valor de filtro (si corresponde); O BIEN
- Una excepción que indica que el URI de etiqueta de EPC no se pudo codificar.

Procedimiento:

1. Use el esquema para identificar la tabla de codificación para este esquema de URI. Si tal esquema no existe, deténgase: este URI no es legal sintácticamente hablando.
2. Confirme que el URI coincida sintácticamente la plantilla de URI asociada con la tabla de codificación. De lo contrario, deténgase: este URI no es legal sintácticamente.
3. Lea la tabla de codificación de izquierda a derecha y construya la codificación especificada en cada columna para obtener una cadena de bits. Si la fila "Cómputo de bits del segmento de codificación" de la tabla especifica una cantidad fija de bits, la cadena de bits obtenida siempre tendrá esta longitud. El método para codificar cada columna depende de la fila "Método de codificación" de la tabla. Si la fila "Método de codificación" especifica una cadena de bits especifica, use esa cadena para esa columna. De lo contrario, consulte las secciones siguientes en las que se especifican los métodos de codificación. Si la codificación de algún segmento falla, deténgase: este URI no se puede codificar.
4. Concatene las cadenas de bits del paso 3 para formar una sola cadena de bits. Si la longitud binaria general que especifica el esquema tiene una longitud fija, entonces la cadena de bits obtenida siempre tendrá esa longitud. La posición de cada segmento en la cadena de bits concatenados se especifica en la fila "Posición binaria" de la tabla de codificación. En la Sección 15.1.1 se especifica el procedimiento que utiliza el resultado de este paso para codificar el banco de memoria de EPC de una etiqueta Gen 2.

En las secciones siguientes se especifican los procedimientos que se usarán en el paso 3.

14.3.1 Método de codificación de "enteros"

El método de codificación de números enteros se utiliza para un segmento que aparece como un entero decimal en el URI, y como un entero binario en la codificación binaria.

Datos ingresados:

Los datos ingresados en el método de codificación es la parte del URI indicada en la fila "Porción del URI" de la tabla de codificación, una cadena de caracteres sin puntos (".").

Prueba de validación:

La cadena de caracteres de entrada debe cumplir con lo siguiente:

- Debe coincidir con la sintaxis de `NumericComponent` según se especifica en la Sección 5.
- El valor de la cadena cuando se considera como un entero decimal debe ser menor que 2^{b}, donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación" de la tabla de codificación.

Si alguna de las pruebas anteriores falla, la codificación del URI falla.

Salida:

La codificación de este segmento es un entero de bits ***b*** (que se llena del lado izquierdo con bits cero según se considere necesario), donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación" de la tabla de codificación, cuyo valor es el valor de la cadena de caracteres de entrada considerado como entero decimal.

14.3.2 Método de codificación de “cadena”

El método de codificación de cadena se utiliza para un segmento que aparece como una cadena alfanumérica en el URI, y como una cadena de bits codificados en ISO 646 (ASCII) en la codificación binaria.

Datos ingresados:

Los datos ingresados en el método de codificación es la parte del URI indicada en la fila “Porción del URI” de la tabla de codificación, una cadena de caracteres sin puntos (".").

Prueba de validación:

La cadena de caracteres de entrada debe cumplir con lo siguiente:

- Debe coincidir con la sintaxis de `GS3A3Component` según se especifica en la Sección 5.
- Para cada porción de la cadena que coincide con la producción de Escape de la sintaxis especificada en la Sección 5 (es decir una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales), los dos caracteres hexadecimales que siguen al carácter % deben relacionarse con uno de los 82 caracteres permitidos que se especifican en la *Tabla A-1*.
- La cantidad de caracteres debe ser igual o menor a ***b***/7, donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación” de la tabla de codificación.

Si alguna de las pruebas anteriores falla, la codificación del URI falla.

Salida:

Considere la entrada como una cadena de cero o más caracteres $S\text{-}S2.SN$, donde cada carácter S_i es un carácter único o una secuencia de 3 caracteres que coincide con la producción de Escape de la sintaxis (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales). Traduzca cada carácter en una cadena de 7 bits. Para un solo carácter, la cadena de 7 bits correspondiente se especifica en la *Tabla A-1*. Para una secuencia de Escape, la cadena de 7 bits es el valor de los dos caracteres hexadecimales considerados como un entero de 7 bits. Concatene esas cadenas de 7 bits en el orden correspondiente a los datos ingresados, luego llene a la derecha con bits cero según sea necesario hasta completar el total de bits ***b***, donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación” de la tabla de codificación. (La cantidad de bits de relleno será ***b*** - 7***N***.) La cadena de bits ***b*** resultante es la salida.

14.3.3 Método de codificación de “tabla de partición”

El método de codificación de tabla de partición se usa para un segmento que aparece en el URI como un par de campos numéricos de longitud variable separados por un carácter de punto (“.”), y en la codificación binaria como un campo de "partición” de 3 bits, seguido de dos enteros binarios de longitud variable. La cantidad de caracteres en los dos campos de URI siempre tiene como resultado una cantidad de caracteres constante, y la cantidad de bits en la codificación binaria tiene como resultado una cantidad constante de bits.

En el método de codificación de la tabla de partición se utiliza una "tabla de partición”. La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

Los datos ingresados en el método de codificación es la porción del URI que se indica en la fila “Porción del URI” de la tabla de codificación. Esto consta de dos cadenas de dígitos separadas por un carácter de punto (“.”). Para los fines de este procedimiento de codificación, las cadenas de dígitos a la izquierda y derecha del punto se denotan como ***C*** y ***D***, respectivamente.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- ***C*** debe coincidir con la sintaxis de `PaddedNumericComponent` según se especifica en la Sección 5.
- ***D*** debe coincidir con la sintaxis de `PaddedNumericComponentOrEmpty` según se especifica en la Sección 5.

- La cantidad de dígitos en ***C*** debe coincidir con uno de los valores especificados en la columna "dígitos del Prefijo GS1 de empresa (L)" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de codificación.
- La cantidad de dígitos en *D* debe coincidir con el valor correspondiente especificado en la otra columna de dígitos del campo de la fila correspondiente de la tabla de partición. Tome en cuenta que si la otra columna de dígitos del campo especifica cero, *D* debe ser la cadena vacía, lo que implica que el segmento de entrada general termina con un carácter de "punto".

Salida:

Desarrolle la cadena binaria de salida concatenando los tres componentes siguientes:

- El valor *P* especificado en la columna "valor de partición" de la fila de la tabla de partición correspondiente, como un entero binario de 3 bits.
- El valor de *C* considerado como un entero decimal, convertido a un entero binario de bits *M*, donde *M* es la cantidad de bits especificada en la columna "bits del Prefijo GS1 de empresa" de la fila de la tabla de partición correspondiente.
- El valor de *D* considerado como un entero decimal, convertido a un entero binario de bits *N*, donde *N* es la cantidad de bits especificada en la otra columna de bits del campo de la fila de la tabla de partición correspondiente. Si *D* es la cadena vacía, el valor del entero de bits *N* es cero.

La cadena de bits resultante es de (3 + *M* + *N*) bits de longitud, lo que siempre equivale al "cómputo de bits del segmento de codificación" de este segmento, según se indica en la tabla de codificación.

14.3.4 Método de codificación de "tabla de partición sin relleno"

El método de codificación de tabla de partición sin relleno se usa para un segmento que aparece en el URI como un par de campos numéricos de longitud variable separados por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits, seguido de dos enteros binarios de longitud variable. La cantidad de caracteres en los dos campos de URI siempre es menor o igual a un límite conocido, y la cantidad de bits en la codificación binaria siempre es una cantidad constante de bits.

En el método de codificación sin relleno de la tabla de partición se utiliza una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

Los datos ingresados en el método de codificación es la porción del URI que se indica en la fila "Porción del URI" de la tabla de codificación. Esto consta de dos cadenas de dígitos separadas por un carácter de punto ("."). Para los fines de este procedimiento de codificación, las cadenas de dígitos a la izquierda y derecha del punto se denotan como ***C*** y ***D***, respectivamente.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- *C* debe coincidir con la sintaxis de `PaddedNumericComponent` según se especifica en la Sección 5.
- *D* debe coincidir con la sintaxis de `NumericComponent` según se especifica en la Sección 5.
- La cantidad de dígitos en *C* debe coincidir con uno de los valores especificados en la columna "dígitos del Prefijo GS1 de empresa (L)" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de codificación.
- El valor de *D* , considerado como un entero decimal, debe ser menor que 2^N, donde *N* es la cantidad de bits especificada en la otra columna de bits del campo de la fila de la tabla de partición correspondiente.

Salida:

Desarrolle la cadena binaria de salida concatenando los tres componentes siguientes:

- El valor *P* especificado en la columna "valor de partición" de la fila de la tabla de partición correspondiente, como un entero binario de 3 bits.

- El valor de *C* considerado como un entero decimal, convertido a un entero binario de bits *M*, donde *M* es la cantidad de bits especificada en la columna "bits del Prefijo GS1 de empresa" de la fila de la tabla de partición correspondiente.
- El valor de *D* considerado como un entero decimal, convertido a un entero binario de bits *N*, donde *N* es la cantidad de bits especificada en la otra columna de bits del campo de la fila de la tabla de partición correspondiente. Si *D* es la cadena vacía, el valor del entero de bits *N* es cero.

La cadena de bits resultante es de (3 + *M* + *N*) bits de longitud, lo que siempre equivale al "cómputo de bits del segmento de codificación" de este segmento, según se indica en la tabla de codificación.

14.3.5 Método de codificación de "tabla de partición de cadena"

El método de codificación de la tabla de partición de cadena se usa para un segmento que aparece en el URI como un campo numérico de longitud variable y un campo de cadena de longitud variable separado por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits seguido de un entero binario de longitud variable y una cadena de caracteres de codificación binaria de longitud variable. La cantidad de caracteres en los dos campos de URI siempre es menor o igual que un límite conocido (computando una secuencia de escape de 3 caracteres como un carácter único) y la cantidad de bits en la codificación binaria se llena si es necesario con una cantidad constante de bits.

En el método de codificación de la tabla de partición se utiliza una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

Los datos ingresados en el método de codificación es la porción del URI que se indica en la fila "Porción del URI" de la tabla de codificación. Esto consta de dos cadenas separadas por un carácter de punto ("."). Para los fines de este procedimiento de codificación, las cadenas a la izquierda y derecha del punto se denotan como ***C*** y ***D***, respectivamente.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- *C* debe coincidir con la sintaxis de `PaddedNumericComponent` según se especifica en la Sección *5*.
- *D* debe coincidir con la sintaxis de `GS3A3Component` según se especifica en la Sección *5*.
- La cantidad de dígitos en *C* debe coincidir con uno de los valores especificados en la columna "dígitos del Prefijo GS1 de empresa (L)" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de codificación.
- La cantidad de caracteres en *D* debe ser menor o igual que el valor correspondiente especificado en la otra columna de caracteres máximos del campo de la fila de la tabla de partición correspondiente. Para los fines de esta regla, un triplete de escape (%nn) se computa como un carácter.
- Para cada porción de *D* que coincide con la producción de Escape de la sintaxis especificada en la Sección *5* (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales), los dos caracteres hexadecimales que siguen al carácter % deben relacionarse con uno de los 82 caracteres permitidos que se especifican en la *Tabla A-1*.

Salida:

Desarrolle la cadena binaria de salida concatenando los tres componentes siguientes:

- El valor *P* especificado en la columna "valor de partición" de la fila de la tabla de partición correspondiente, como un entero binario de 3 bits.
- El valor de *C* considerado como un entero decimal, convertido a un entero binario de bits *M*, donde *M* es la cantidad de bits especificada en la columna "bits del Prefijo GS1 de empresa" de la fila de la tabla de partición correspondiente.
- El valor de *D* convertido en una cadena binaria de bits *N*, donde *N* es la cantidad de bits especificada en la otra columna de bits del campo de la fila de la tabla de partición correspondiente. Esta cadena binaria de bits *N* se construye como se muestra a continuación. Considere *D* como una cadena de cero o más caracteres $S_1S_2...S_N$, donde cada carácter S_i es un carácter único o una secuencia de 3 caracteres que coincide con la producción de Escape de la sintaxis (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales). Traduzca cada carácter a una cadena de 7 bits. Para un carácter simple, la cadena correspondiente de 7 bits se especifica en la Tabla A-1. Para una secuencia de *Escape,* la cadena de 7 bits es el valor de los dos caracteres hexadecimales considerados como un entero de 7 bits. Concatene esas cadenas de 7 bits en el orden correspondiente a los datos ingresados, y luego rellene con bits de cero según sea necesario para llegar a un total de *N* bits.

  La cadena resultante de bits tiene (3 + M + N) bits de longitud, que siempre equivale al "Conteo de bits del segmento de codificación" de este segmento según se indica en la tabla de codificación.

14.3.6 Método de codificación de “cadena numérica”

El método de codificación de cadena numérica se usa para un segmento que aparece como una cadena numérica en el URI, posiblemente incluidos los ceros iniciales. Los ceros iniciales se conservan en la codificación binaria prefijando un dígito “1” a la cadena numérica antes de la codificación.

Datos ingresados:

Los datos ingresados en el método de codificación es la parte del URI indicada en la fila “Porción del URI” de la tabla de codificación, una cadena de caracteres sin puntos (".").

Prueba de validación:

La cadena de caracteres de entrada debe cumplir con lo siguiente:

- Debe coincidir con la sintaxis de `PaddedNumericComponent` según se especifica en la Sección ***5***.
- La cantidad de dígitos en la cadena, D, debe ser tal que $2 \times 10^{D} < 2^{b}$, donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación” de la tabla de codificación. (Para el esquema GDTI-113, ***b*** = 58 y por ende la cantidad de dígitos D debe ser menor o igual que 17. GDTI-113 y SGCN-96 son los únicos esquemas que utilizan este método de codificación).

Si alguna de las pruebas anteriores falla, la codificación del URI falla.

Salida:

Construya la cadena de bits de salida como se indica a continuación:

- Prefije el carácter "1” a la izquierda de la cadena de caracteres de entrada.
- Convierta la cadena resultante a un entero de bits ***b*** (que se llena del lado izquierdo con bits cero según se considere necesario), donde ***b*** es el valor especificado en la fila "Cómputo de bits" de la tabla de codificación, cuyo valor es el valor de la cadena de caracteres de entrada considerado como entero decimal.

14.3.7 Método de codificación “CAGE/DODAAC de 6 bits”

El método de codificación CAGE/DoDAAC de 6 bits se usa para un segmento que aparece como un código CAGE de 5 caracteres o DoDAAC de 6 caracteres en el URI, y como una cadena de bits codificada de 36 bits en la codificación binaria.

Datos ingresados:

Los datos ingresados en el método de codificación es la parte del URI indicada en la fila “Porción del URI” de la tabla de codificación, una cadena de 5 o 6 caracteres sin puntos (".").

Prueba de validación:

La cadena de caracteres de entrada debe cumplir con lo siguiente:

- Debe coincidir con la sintaxis de `CAGECodeOrDODAAC` según se especifica en la Sección *6.3.14*.

Si la prueba anterior falla, la codificación del URI falla.

Salida:

Considere los datos ingresados como una cadena de cinco o seis caracteres *d1d2...dN*, donde cada carácter d_i es un carácter único. Traduzca cada carácter en una cadena de 6 bits con la *Tabla G-1* (*G*). Concatene esas cadenas de 6 bits en el orden correspondiente de los datos ingresados. Si los datos ingresados son cinco caracteres, prefije el valor de 6 bits 100000 a la izquierda del resultado. La cadena resultante de 36 bits es la salida.

14.3.8 Método de codificación de "cadena variable de 6 bits"

El método de codificación de cadena variable de 6 bits se usa para un segmento que aparece en el URI como un campo de cadena, y en la codificación binaria como una cadena de caracteres de codificación binaria que termina en un valor nulo, de longitud variable.

Datos ingresados:

Los datos ingresados en el método de codificación es la porción del URI que se indica en la fila "Porción del URI" de la tabla de codificación.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los datos ingresados deben coincidir con la sintaxis de la porción correspondiente del URI según se especifica en la subsección adecuada de la Sección *6.3*.
- La cantidad de caracteres en la entrada debe ser mayor o igual que la cantidad mínima de caracteres y menor o igual que la cantidad máxima de caracteres especificada en la nota al pie de la tabla de codificación para esta columna de la tabla de codificación. Para los fines de esta regla, un triplete de escape (%nn) se computa como un carácter.
- Para cada porción de la entrada que coincide con la producción de *Escape* de la sintaxis especificada en la Sección *5* (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales), los dos caracteres hexadecimales que siguen al carácter % deben relacionarse con uno de los caracteres permitidos que se especifican en la *Tabla G-1* (*G*), y el carácter correlacionado debe cumplir con cualquier otra restricción especificada en la tabla de codificación para este segmento de codificación.
- Para cada porción de la entrada que sea un carácter único (contrario a una secuencia de escape de 3 caracteres), ese carácter debe cumplir con cualquier otra restricción especificada en la tabla de codificación para este segmento de codificación.

Salida:

Considere la entrada como una cadena de cero o más caracteres *s1S2...sn*, donde cada carácter S_i es un carácter único o una secuencia de 3 caracteres que coincide con la producción de Escape de la sintaxis (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales). Traduzca cada carácter a una cadena de 6 bits. Para un solo carácter, la cadena de 6 bits correspondiente se especifica en la *Tabla G-1* (*G*). Para una secuencia de Escape, la cadena de 6 bits correspondiente se especifica en la in *Tabla G-1* (*G*) identificando la secuencia de escape en la columna "Forma de URI". Concatene esas cadenas de 6 bits en el orden correspondiente de los datos ingresados, luego prefije seis bits cero (000000).

La cadena de bits resultante es de longitud variable, pero siempre de por lo menos 6 bits y siempre es un múltiplo de 6 bits.

14.3.9 Método de codificación de "tabla de partición de cadena variable de 6 bits"

El método de codificación de la tabla de partición de cadena variable de 6 bits se usa para un segmento que aparece en el URI como un campo numérico de longitud variable y un campo de cadena de longitud variable separado por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits seguido de un entero binario de longitud variable y una cadena de caracteres de codificación binaria que termina en un valor nulo. La cantidad de caracteres en los dos campos de URI siempre es menor o igual que un límite conocido (computando una secuencia de escape de 3 caracteres como un carácter único) y la cantidad de bits en la codificación binaria también es menor o igual que un límite conocido.

En el método de codificación de tabla de partición de cadena variable de 6 bits se usa una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

Los datos ingresados en el método de codificación es la porción del URI que se indica en la fila "Porción del URI" de la tabla de codificación. Esto consta de dos cadenas separadas por un carácter de punto (".") Para los fines de este procedimiento de codificación, las cadenas a la izquierda y derecha del punto se denotan como ***C*** y ***D***, respectivamente.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los datos ingresados deben coincidir con la sintaxis de la porción correspondiente del URI según se especifica en la subsección adecuada de la Sección 6.3.
- La cantidad de dígitos en *C* debe coincidir con uno de los valores especificados en la columna "dígitos del Prefijo GS1 de empresa (L)" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de codificación.
- La cantidad de caracteres en *D* debe ser menor o igual que el valor correspondiente especificado en la otra columna de caracteres máximos del campo de la fila de la tabla de partición correspondiente. Para los fines de esta regla, un triplete de escape (%nn) se computa como un carácter.
- Para cada porción de *D* que coincide con la producción de `Escape` de la sintaxis especificada en la Sección 5 (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales), los dos caracteres hexadecimales que siguen al carácter % deben relacionarse con uno de los 39 caracteres permitidos que se especifican en la *Tabla G-1* (*G*).

Salida:

Desarrolle la cadena binaria de salida concatenando los tres componentes siguientes:

- El valor *P* especificado en la columna "valor de partición" de la fila de la tabla de partición correspondiente, como un entero binario de 3 bits.
- El valor de *C* considerado como un entero decimal, convertido a un entero binario de bits *M*, donde *M* es la cantidad de bits especificada en la columna "bits del Prefijo GS1 de empresa" de la fila de la tabla de partición correspondiente.
- El valor de *D* convertido en una cadena binaria de bits *N*, donde *N* es menor o igual que la cantidad de bits especificada en la otra columna de bits del campo de la fila de la tabla de partición correspondiente. Esta cadena binaria se construye como se muestra a continuación. Considere *D* como una cadena de uno o más caracteres S_1-*S*2.SN, donde cada carácter *S*i es un carácter único o una secuencia de 3 caracteres que coincide con la producción de Escape de la sintaxis (es decir, una secuencia de 3 caracteres que consta de un carácter % seguido de dos dígitos hexadecimales). Traduzca cada carácter a una cadena de 6 bits. Para un solo carácter, la cadena de 6 bits correspondiente se especifica en la *Tabla G-1* (*G*). Para una secuencia de Escape, la cadena de 6 bits es el valor de los dos caracteres hexadecimales considerados como un entero de 6 bits. Concatene esas cadenas de 6 bits en el orden correspondiente de los datos ingresados, luego añada seis bits cero.

La cadena de bits resultante es de (3 + *M* + *N*) bits de longitud, lo que siempre es menor o igual que el "cómputo máximo de bits del segmento de codificación" de este segmento, según se indica en la tabla de codificación.

14.3.10 Método de codificación de "entero de ancho fijo"

El método de codificación de números enteros de ancho fijo se utiliza para un segmento que aparece como un entero decimal llenado con ceros en el URI, y como un entero binario en la codificación binaria.

Datos ingresados:

Los datos ingresados en el método de codificación es la parte del URI indicada en la fila "Porción del URI" de la tabla de codificación, una cadena de caracteres numérica sin puntos (".").

Prueba de validación:

La cadena de caracteres de entrada debe cumplir con lo siguiente:

- Debe coincidir con la sintaxis de `PaddedNumericComponent` según se especifica en la Sección 5.

- El valor de la cadena cuando se considera un entero decimal no negativo debe ser menor que ((10^A D) -1) donde D=int(b*log(2)/log(10)), donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación" de la tabla de codificación.

Si alguna de las pruebas anteriores falla, la codificación del URI falla.

Salida:

La codificación de este segmento es un entero de bits ***b*** (que se llena del lado izquierdo con bits cero según se considere necesario), donde ***b*** es el valor especificado en la fila "Cómputo de bits del segmento de codificación" de la tabla de codificación, cuyo valor es el valor de la cadena de caracteres de entrada considerado como entero decimal.

14.4 Procedimiento de decodificación

Este procedimiento codifica una cadena de bits que inicia con el bit 20_h en el banco de memoria de EPC de una etiqueta Gen 2 en un URI de etiqueta de EPC. Este procedimiento solo codifica el valor de EPC y del filtro (si corresponde). La sección *15.2.2* proporciona el procedimiento completo para decodificar el contenido completo del banco de memoria de EPC, incluida la información de control que se guarda afuera del EPC codificado. El procedimiento de la Sección *15.2.2* se debe usar en la mayoría de las aplicaciones. (El procedimiento en la Sección *15.2.2* utiliza el procedimiento siguiente como subrutina.)

Datos:

- Una cadena de bits consta de bits N $\boldsymbol{b}_{N-1}\boldsymbol{b}_{N-2}\ldots\boldsymbol{b}_0$

Resultado:

- Un URI de etiqueta de EPC que comienza con `urn:epc:tag:`, que no contiene campos de información de control (además del valor de filtro si el esquema de EPC incluye un valor de filtro); O BIEN
- Una excepción que indica que la cadena de bits no se puede decodificar en un URI de etiqueta de EPC.

Procedimiento:

1. Extraiga los ocho bits más significativos, el encabezado de EPC: $b_{N-1}b_{N-2}\ldots b_{N-8}$. Consulte la *Tabla 14-1* en la Sección *14.2*, y use el encabezado para identificar la tabla de codificación para esta codificación binaria y la longitud del bit de codificación *B*. Si no existe ninguna tabla de codificación para este encabezado, deténgase: esta codificación binaria no se puede decodificar.
2. Confirme que la cantidad total de bits *N* es mayor o igual que la cantidad total de bits *B* especificada para este encabezado en la *Tabla 14-1*. Si no lo es, deténgase: esta codificación binaria no se puede decodificar.
3. De ser necesario, trunque los bits menos significativos de la entrada para que coincidan con la cantidad de bits especificada en la *Tabla 14-1*; es decir, si en la *Tabla 14-1* se especifican los bits *B*, retenga los bits $b_{N-1}b_{N-2}\ldots b_{N-B}$. Para el resto de este procedimiento, considere que los bits restantes se enumeraron como se muestra a continuación: $b_{B-1}b_{B-2}\ldots b_0$. (El objetivo de este paso es eliminar cualquier bit de relleno de ceros iniciales que se puedan haber interpretado debido a una transferencia de datos orientada a las palabras).
4. Para un esquema de codificación de longitud variable, no se ha especificado ningún bit *B* en la *Tabla 14-1* por lo que este paso se debe omitir. Es posible que sobren bits de relleno de ceros iniciales después de que todos los segmentos se decodifiquen en el paso 4 más adelante; si eso sucede, ignórelos.
5. Separe los bits de la codificación binaria en segmentos, de acuerdo con la fila "posición binaria" de la tabla de codificación. Para cada segmento, decodifique los bits para obtener una cadena de caracteres que se usará como una porción del URI final. El método para decodificar cada columna depende de la fila "Método de codificación" de la tabla. Si la fila "Método de codificación" especifica una cadena de bits especifica, los bits correspondientes deben coincidir con esos bits exactamente; de lo contrario, deténgase: esta codificación binaria no se puede decodificar. De lo contrario, consulte las secciones siguientes en las que se especifican los métodos de decodificación. Si la decodificación de algún segmento falla, deténgase: esta codificación binaria no se puede decodificar.
6. Para un segmento de codificación de longitud variable, el método de codificación se aplica comenzando con el bit que sigue a los bits que consumió la columna de codificación previa. Es decir, si la columna de codificación previa (la columna a la izquierda de esta) consumió bits hasta el bit b_i, el bit más significativo para decodificar este segmento es el bit b_{i-1}. El método de codificación determinará el lugar donde se localizará el bit final de este segmento.

7. Concatene las cadenas siguientes para obtener el URI final: la cadena urn:epc:tag:, el nombre del esquema según se especificó en la tabla de codificación, un carácter de dos puntos (":") y las cadenas obtenidas en el paso 4, inserte un carácter de punto (".") entre las cadenas adyacentes.

En las secciones siguientes se especifican los procedimientos que se usarán en el paso 4.

14.4.1 Método de decodificación de "enteros"

El método de decodificación de números enteros se utiliza para un segmento que aparece como un entero decimal en el URI, y como un entero binario en la codificación binaria.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación.

Prueba de validación:

No hay pruebas de validación para este método de decodificación.

Salida:

La decodificación de este segmento es un numeral decimal cuyo valor es el valor de la entrada considerado como un entero binario sin signo. La salida no debe comenzar con un carácter de cero si tiene dos o más dígitos de longitud.

14.4.2 Método de decodificación de "cadena"

El método de decodificación de cadena se utiliza para un segmento que aparece como una cadena alfanumérica en el URI, y como una cadena de bits codificados en ISO 646 (ASCII) en la codificación binaria.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación. Esta longitud de esta cadena de bits siempre es un múltiplo de siete.

Prueba de validación:

La cadena de bits de entrada debe cumplir con lo siguiente:

- Cada segmento de 7 bits debe tener un valor que corresponda a un carácter especificado en la *Tabla A-1*, o todos los valores deben ser ceros.
- Todos los segmentos de 7 bits que aparezcan después de un segmento donde todos los valores sean ceros también deben ser ceros.
- El primer segmento de 7 bits no debe tener puros ceros. (En otras palabras, la cadena debe contener por lo menos un carácter).

Si alguna de las pruebas anteriores falla, la decodificación del segmento falla.

Salida:

Traduzca cada segmento de 7 bits, hasta el segmento con todos los valores cero (si lo hay), en un carácter único o en un triplete escape de 3 caracteres, busque el segmento de 7 bits en la *Tabla A-1*, y use el valor encontrado en la columna "Forma de URI". Concatene los caracteres y/o los tripletes de 3 caracteres en el orden que corresponda a la cadena de bits de entrada. La cadena de caracteres resultante es la salida. Esta cadena de caracteres coincide con la producción de GS3A3 de la sintaxis en la Sección 5.

14.4.3 Método de decodificación de "tabla de partición"

El método de decodificación de tabla de partición se usa para un segmento que aparece en el URI como un par de campos numéricos de longitud variable separados por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits, seguido de dos enteros binarios de longitud variable. La cantidad de caracteres en los dos campos de URI siempre tiene como resultado una cantidad de caracteres constante, y la cantidad de bits en la codificación binaria tiene como resultado una cantidad constante de bits.

En el método de decodificación de la tabla de partición se utiliza una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación. En términos lógicos, esta cadena de bits se divide en tres subcadenas, que constan de un valor de "partición" de 3 bits, seguido de dos subcadenas de longitud variable.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los tres bits más significativos de la cadena de bits de entrada, considerados como un entero binario, deben coincidir con uno de los valores especificados en la columna de "valor de partición" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de decodificación.
- Extraiga la *M* que aparece al lado de los bits más significativos de la cadena de bits de entrada después de los tres bits de partición, donde *M* es el valor especificado en la columna "Bits del Prefijo de empresa" de la fila de la tabla de partición correspondiente. Considere estos bits *M* como un entero binario sin signo, *C*. El valor de *C* debe ser menor que 10^L, donde *L* es el valor especificado en la columna "Dígitos del Prefijo GS1 de empresa (L)" de la fila de la tabla de partición correspondiente.
- Hay bits *N* restantes en la cadena de bits de entrada, donde *N* es el valor especificado en la otra columna de bits de campo de la fila de la tabla de partición correspondiente. Considere estos bits *N* como un entero binario sin signo, *D*. El valor de *D* debe ser menor que 10^K, donde *K* es el valor especificado en la otra columna (K) de dígitos del campo de la fila de la tabla de partición correspondiente. Tome en cuenta que si $\boldsymbol{K = 0}$, el valor de ***D*** debe ser cero.

Salida:

Desarrolle la cadena de caracteres de salida concatenando los tres componentes siguientes:

- El valor *C* convertido en un numeral decimal, llenado a la izquierda con caracteres cero ("0") para crear dígitos *L* en total.
- Un carácter de punto (".").
- El valor *C* convertido en un numeral decimal, llenado a la izquierda con caracteres cero ("0") para crear dígitos *L* en total. Si *K* = 0, no prefije ningún carácter al punto de arriba (en este caso, la cadena de URI final tendrá dos caracteres de punto adyacentes cuando este segmento se combine con el segmento siguiente).

14.4.4 Método de decodificación de "tabla de partición sin relleno"

El método de decodificación de tabla de partición sin relleno se usa para un segmento que aparece en el URI como un par de campos numéricos de longitud variable separados por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits, seguido de dos enteros binarios de longitud variable. La cantidad de caracteres en los dos campos de URI siempre es menor o igual a un límite conocido, y la cantidad de bits en la codificación binaria siempre es una cantidad constante de bits.

En el método de decodificación sin relleno de la tabla de partición se utiliza una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación. En términos lógicos, esta cadena de bits se divide en tres subcadenas, que constan de un valor de "partición" de 3 bits, seguido de dos subcadenas de longitud variable.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los tres bits más significativos de la cadena de bits de entrada, considerados como un entero binario, deben coincidir con uno de los valores especificados en la columna de "valor de partición" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de decodificación.
- Extraiga la *M* que aparece al lado de los bits más significativos de la cadena de bits de entrada después de los tres bits de partición, donde *M* es el valor especificado en la columna "Bits del Prefijo de empresa" de la fila de la tabla de partición correspondiente. Considere estos bits *M* como un entero binario sin signo, *C*. El valor de *C* debe ser menor que 10^L, donde *L* es el valor especificado en la columna "Dígitos del Prefijo GS1 de empresa (L)" de la fila de la tabla de partición correspondiente.
- Hay bits *N* restantes en la cadena de bits de entrada, donde *N* es el valor especificado en la otra columna de bits de campo de la fila de la tabla de partición correspondiente. Considere estos bits *N* como un entero binario sin signo, *D*.

Salida:

Desarrolle la cadena de caracteres de salida concatenando los tres componentes siguientes:

- El valor *C* convertido en un numeral decimal, llenado a la izquierda con caracteres cero ("0") para crear dígitos *L* en total.
- Un carácter de punto (".").
- El valor *D* convertido en un numeral decimal, sin ceros iniciales (salvo que si *D* = 0, se convierte en un dígito cero único).

14.4.5 Método de decodificación de "tabla de partición de cadena"

El método de decodificación de la tabla de partición de cadena se usa para un segmento que aparece en el URI como un campo numérico de longitud variable y un campo de cadena de longitud variable separado por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits seguido de un entero binario de longitud variable y una cadena de caracteres de codificación binaria de longitud variable. La cantidad de caracteres en los dos campos de URI siempre es menor o igual que un límite conocido (computando una secuencia de escape de 3 caracteres como un carácter único) y la cantidad de bits en la codificación binaria se llena si es necesario con una cantidad constante de bits.

En el método de decodificación de la tabla de partición se utiliza una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación. En términos lógicos, esta cadena de bits se divide en tres subcadenas, que constan de un valor de "partición" de 3 bits, seguido de dos subcadenas de longitud variable.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los tres bits más significativos de la cadena de bits de entrada, considerados como un entero binario, deben coincidir con uno de los valores especificados en la columna de "valor de partición" de la tabla de partición. La fila correspondiente se llama "fila de la tabla de partición correspondiente" en el resto del procedimiento de decodificación.
- Extraiga la *M* que aparece al lado de los bits más significativos de la cadena de bits de entrada después de los tres bits de partición, donde *M* es el valor especificado en la columna "Bits del Prefijo de empresa" de la fila de la tabla de partición correspondiente. Considere estos bits *M* como un entero binario sin signo, *C*. El valor de *C* debe ser menor que 10^L, donde *L* es el valor especificado en la columna "Dígitos del Prefijo GS1 de empresa (L)" de la fila de la tabla de partición correspondiente.
- Hay bits *N* restantes en la cadena de bits de entrada, donde *N* es el valor especificado en la otra columna de bits de campo de la fila de la tabla de partición correspondiente. Estos bits deben constar de uno o más segmentos de 7 bits sin ceros, seguidos de ceros o más bits donde todos los valores son ceros.
- La cantidad de segmentos de 7 bits sin ceros que precede a los bits de ceros (si los hay) debe ser menor o igual que *K*, donde *K* es el valor especificado en la columna "Máximo de caracteres" de la fila de la tabla de partición correspondiente.

- Cada uno de los segmentos de 7 bits sin ceros debe tener un valor correspondiente a un carácter especificado en la *Tabla A-1*.

Salida:

Desarrolle la cadena de caracteres de salida concatenando los tres componentes siguientes:

- El valor *C* convertido en un numeral decimal, llenado a la izquierda con caracteres cero ("0") para crear dígitos *L* en total.
- Un carácter de punto (".").
- Una cadena de caracteres que se determina como se indica a continuación. Traduzca cada segmento de 7 bits sin ceros según lo determine la prueba de validez en un carácter único o en un triplete de escape de 3 caracteres identificando el segmento de 7 bits en la *Tabla A-1*, y usando el valor que se encuentra en la columna "Forma de URI". Concatene los caracteres y/o los tripletes de 3 caracteres en el orden que corresponda a la cadena de bits de entrada.

14.4.6 Método de decodificación de "cadena numérica"

El método de decodificación de cadena numérica se usa para un segmento que aparece como una cadena numérica en el URI, posiblemente incluidos los ceros iniciales. Los ceros iniciales se conservan en la codificación binaria prefijando un dígito "1" a la cadena numérica antes de la codificación.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación.

Prueba de validación:

La entrada debe ser tal que el procedimiento de decodificación a continuación no falle.

Salida:

Construya la cadena de salida como se indica a continuación.

- Convierta la cadena de bits de entrada en un numeral decimal sin ceros iniciales cuyo valor es el valor de la entrada considerado como un entero binario sin signos.
- Si el numeral del paso previo no comienza con un carácter "1", deténgase: la entrada es inválida.
- Si el numeral del paso previo consta únicamente de un carácter, deténgase: la entrada es inválida (ya que esto correspondería a una cadena numérica vacía).
- Elimine el carácter "1" inicial del numeral.
- La cadena resultante es la salida.

14.4.7 Método de decodificación "CAGE/DoDAAC de 6 bits"

El método de decodificación CAGE/DoDAAC de 6 bits se usa para un segmento que aparece como un código CAGE de 5 caracteres o un código DoDAAC de 6 caracteres en el URI, y como una cadena de bits codificada de 36 bits en la codificación binaria.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila "posición binaria" de la tabla de codificación. La longitud de esta cadena de bits siempre es 36 bits.

Prueba de validación:

La cadena de bits de entrada debe cumplir con lo siguiente:

- Cuando se considera que la cadena de bits consta de seis segmentos de 6 bits, cada segmento de 6 bits debe tener un valor que corresponde a un carácter especificado en la *Tabla G-1* (*G*) salvo que el primer segmento de 6 bits también puede tener el valor de 100000.

- El primer segmento de 6 bits debe tener el valor de 100000, o corresponder a un carácter de dígito, o un carácter alfabético de letra mayúscula, excepto por las letras I y O.
- Los segmentos de 6 bits restantes deben corresponder a un carácter de dígito o un carácter alfabético de letra mayúscula, excepto por las letras I y O.

Si alguna de las pruebas anteriores falla, la decodificación del segmento falla.

Salida:

Deseche el primer segmento de 6 bits si equivale a 100000. Traduzca cada uno de los cinco o seis segmentos de 6 bits restantes en un carácter único buscando el segmento de 6 bits en la *Tabla G-1* (*G*) y usando el valor que se encuentra en la columna "Forma de URI". Concatene los caracteres en el orden que corresponde a la cadena de bits de entrada. La cadena de caracteres resultante es la salida. Esta cadena de caracteres coincide con la producción de CAGECodeOrDODAAC de la sintaxis en la Sección *6.3.14*.

14.4.8 Método de decodificación de "cadena variable de 6 bits"

El método de decodificación de cadena variable de 6 bits se usa para un segmento que aparece en el URI como un campo de cadena de longitud variable, y en la codificación binaria como una cadena de caracteres de codificación binaria que termina en un valor nulo, de longitud variable.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits que comienza en la siguiente posición binaria menos significativa después del segmento de codificación previo. Este método de decodificación solo consume una porción de esta cadena de bits, según se describe a continuación.

Prueba de validación:

La entrada debe ser tal que el procedimiento de decodificación a continuación no falle.

Salida:

Construya la cadena de salida como se indica a continuación.

- Comenzando con el bit más significativo de la entrada, divida la entrada en segmentos de 6 bits adyacentes, hasta que se encuentre un segmento de terminación que conste de puros bits cero (000000). Si la entrada se agota antes de que se encuentre el segmento de puros ceros, deténgase: la entrada es inválida.
- La cantidad de segmentos de 6 bits que preceden al segmento de terminación debe ser mayor o igual que la cantidad mínima de caracteres y menor o igual que la cantidad máxima de caracteres especificada en la nota al pie de la tabla de codificación para esta columna de la tabla de codificación. De lo contrario, deténgase: la entrada es inválida.
- Para cada segmento de 6 bits que precede al segmento de terminación, consulte la *Tabla G-1* (*G*) para encontrar el carácter que corresponde al valor del segmento de 6 bits. Si no hay ningún carácter en la tabla que corresponda al segmento de 6 bits, deténgase: la entrada es inválida.
- Si la entrada viola cualquier otra restricción indicada en la tabla de codificación, deténgase: la entrada es inválida.
- Traduzca cada segmento de 6 bits que preceda al segmento de terminación en un carácter único o en un tripleteo de escape de 3 caracteres identificando el segmento de 6 bits en la *Tabla G-1* (*G*) y usando el valor que se encuentra en la columna "Forma de URI". Concatene los caracteres y/o los tripletes de 3 caracteres en el orden que corresponda a la cadena de bits de entrada. La cadena resultante es la salida del procedimiento de decodificación.
- Si las columnas permanecen en la tabla de codificación, el procedimiento de decodificación para la columna siguiente se reanuda con el siguiente bit menos significativo después del segmento de terminación 000000.

14.4.9 Método de decodificación de "tabla de partición de cadena variable de 6 bits"

El método de decodificación de la tabla de partición de cadena variable de 6 bits se usa para un segmento que aparece en el URI como un campo numérico de longitud variable y un campo de cadena de longitud variable separado por un carácter de punto ("."), y en la codificación binaria como un campo de "partición" de 3 bits seguido de un entero binario de longitud variable y una cadena de caracteres de codificación binaria que termina en un valor nulo. La cantidad de caracteres en los dos campos de URI siempre es menor o igual que un límite conocido (computando una secuencia de escape de 3 caracteres como un carácter único) y la cantidad de bits en la codificación binaria también es menor o igual que un límite conocido.

En el método de decodificación de tabla de partición de cadena variable de 6 bits se usa una "tabla de partición". La tabla de partición específica que se utiliza se especifica en la tabla de codificación para un esquema de EPC determinado.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila “posición binaria” de la tabla de codificación. En términos lógicos, esta cadena de bits se divide en tres subcadenas, que constan de un valor de “partición” de 3 bits, seguido de dos subcadenas de longitud variable.

Prueba de validación:

La entrada debe cumplir con lo siguiente:

- Los tres bits más significativos de la cadena de bits de entrada, considerados como un entero binario, deben coincidir con uno de los valores especificados en la columna de “valor de partición” de la tabla de partición. La fila correspondiente se llama “fila de la tabla de partición correspondiente” en el resto del procedimiento de decodificación.
- Extraiga la *M* que aparece al lado de los bits más significativos de la cadena de bits de entrada después de los tres bits de partición, donde *M* es el valor especificado en la columna "Bits del Prefijo de empresa” de la fila de la tabla de partición correspondiente. Considere estos bits *M* como un entero binario sin signo, *C*. El valor de *C* debe ser menor que 10^L, donde *L* es el valor especificado en la columna "Dígitos del Prefijo GS1 de empresa (L)” de la fila de la tabla de partición correspondiente.
- Hay un límite máximo de *N* bits restantes en la cadena de bits de entrada, donde *N* es el valor especificado en la otra columna de cantidad máxima de bits de campo de la fila de la tabla de partición correspondiente. Estos bits deben comenzar con uno o más segmentos de 6 bits sin ceros, seguidos de seis bits de ceros. Cualquier bit adicional después de los seis bits de ceros pertenecen al siguiente segmento de codificación en la tabla de codificación.
- La cantidad de segmentos de 6 bits sin ceros que precede a los bits de ceros debe ser menor o igual que *K*, donde *K* es el valor especificado en la columna "Máximo de caracteres” de la fila de la tabla de partición correspondiente.
- Cada uno de los segmentos de 6 bits sin ceros debe tener un valor correspondiente a un carácter especificado en la *Tabla G-1* (*G*)

Salida:

Desarrolle la cadena de caracteres de salida concatenando los tres componentes siguientes:

- El valor *C* convertido en un numeral decimal, llenado a la izquierda con caracteres cero ("0”) para crear dígitos *L* en total.
- Un carácter de punto (".”).
- Una cadena de caracteres que se determina como se indica a continuación. Traduzca cada segmento de 6 bits sin ceros según lo determine la prueba de validez en un carácter único o en un triplete de escape de 3 caracteres identificando el segmento de 6 bits en la *Tabla G-1* (*G*) y usando el valor que se encuentra en la columna "Forma de URI”. Concatene los caracteres y/o los tripletes de 3 caracteres en el orden que corresponda a la cadena de bits de entrada.

14.4.10 Método de decodificación de “entero de ancho fijo”

El método de decodificación de números enteros se utiliza para un segmento que aparece como un entero decimal llenado con ceros en el URI, y como un entero binario en la codificación binaria.

Datos ingresados:

La entrada del método de decodificación es la cadena de bits identificada en la fila “posición binaria” de la tabla de codificación.

Prueba de validación:

Según una secuencia de bits de longitud b, calcule $i_{máx.}$ como se indica a continuación:

D = int(b*log(2)/log(10))

$i_{máx.}$ = 10^D — 1

Interprete la secuencia de bits de longitud b como un valor entero que no es negativo, i

Si i > $i_{máx.}$, entonces la decodificación falla porque los bits corresponden a un valor que no se puede expresar en dígitos D.

Salida:

La decodificación de este segmento es un numeral decimal cuyo valor es el valor de la entrada considerado como un entero binario sin signo. La salida se llena a la izquierda, de modo que la cantidad total de dígitos D se determina calculando D=int(b*log(2)/log(10)).

14.5 Tablas de codificación binaria de EPC

En esta sección se especifican las tablas de codificación que se usan con el procedimiento de codificación de la Sección *14.3* y el procedimiento de decodificación de la Sección *14.3.4*.

La fila "Posición binaria" de cada tabla de codificación ilustra las posiciones binarias relativas en cada codificación binaria. En la fila "Posición binaria", el subíndice más alto indica el bit más significativo, y el subíndice 0 indica el bit menos significativo. Tome en cuenta que esto es lo opuesto a la manera en que las direcciones de bit del banco de memoria de etiquetas RFID se indican normalmente, donde la dirección 0 es el bit más significativo.

14.5.1 Número global de artículo comercial serializado (SGTIN)

Se especifican dos esquemas de codificación para el SGTIN, una codificación de 96 bits (SGTIN-96) y una codificación de 198 bits (SGTIN-198). La codificación SGTIN-198 permite el rango completo de números de serie hasta por 20 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación SGTIN-96 permite los números de serie únicamente numéricos, sin ceros iniciales, cuyo valor es menor que 2^{38} (es decir, de 0 a 274,877,906,943 inclusive).

Ambos esquemas de codificación de SGTIN hacen referencia a la tabla de partición siguiente.

Tabla 14-2 Tabla de partición de SGTIN

Valor de la partición *(P)*	Prefijo GS1 de empresa		Indicador/dígito de relleno y referencia del artículo	
	Bits ***(M)***	**Dígitos** ***(L)***	**Bits** **(N)**	**Dígitos**
0	40	12	4	1
1	37	11	7	2
2	34	10	10	3
3	30	9	14	4
4	27	8	17	5
5	24	7	20	6
6	20	6	24	7

14.5.1.1 Tabla de codificación SGTIN-96

Tabla 14-3 Tabla de codificación SGTIN-96

Esquema	SGTIN-96
Plantilla de URI	`urn:epc:tag:sgtin-96:`*`F.C.I.S`*

Esquema	SGTIN-96					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa (*)	Indicador (**) / Referencia del artículo	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	38
Segmento de codificación	Encabezado de EPC	Filtro	GTIN			Seriado
Porción del URI		*F*	*C.I*			*S*
Cómputo de bits del segmento de codificación	8	3	47			38
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{38}$			$b_{37}b_{36}\ldots b_{0}$
Método de codificación	00110000	Entero	Partición *Tabla 14-2*			Entero

(*) Vea la Sección 7.1.2 para conocer el caso de un SGTIN derivado de GTIN-8.

(**) Tome en cuenta que en el caso de un SGTIN derivado de un GTIN-12 o GTIN-13, un dígito cero de relleno toma el lugar del dígito indicador. En todos los casos, vea la Sección 7.1 para conocer la definición de la manera en que el dígito indicado (o dígito cero de relleno) y la referencia del artículo se combinan en este segmento del EPC.

14.5.1.2 Tabla de codificación SGTIN-198

Tabla 14-4 Tabla de codificación SGTIN-198

Esquema	SGTIN-198					
Plantilla de URI	urn:epc:tag:sgtin-i98:*F.C.I.S*					
Total de bits	198					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa (*)	Indicador (**) / Referencia del artículo	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	140
Segmento de codificación	Encabezado de EPC	Filtro	GTIN			Seriado
Porción del URI		*F*	*C.I*			*S*
Cómputo de bits del segmento de codificación	8	3	47			140
Posición binaria	***b***197***b***196…***b***190	***b***189***b***188***b***187	***b***186***b***185…***b***140			***b***139***b***138…***b***0
Método de codificación	00110110	Entero	Partición ***Tabla 14-2***			Secuencia

(*) Vea la Sección 7.1.2 para conocer el caso de un SGTIN derivado de GTIN-8.

(**) Tome en cuenta que en el caso de un SGTIN derivado de un GTIN-12 o GTIN-13, un dígito cero de relleno toma el lugar del dígito indicador. En todos los casos, vea la Sección 7.1 para conocer la definición de la manera en que el dígito indicado (o dígito cero de relleno) y la referencia del artículo se combinan en este segmento del EPC.

14.5.2 Código seriado de contenedor de envío (SSCC)

Se especifica un esquema de codificación para el SSCC: el SSCC-96 de codificación de 96 bits. La codificación SSCC-96 permite el rango completo de SSCC, según se especifica en [GS1GS1].

El esquema de codificación SSCC-96 hace referencia a la tabla de partición siguiente.

Tabla 14-5 Tabla de partición de SSCC

Valor de la partición *(P)*	Prefijo GS1 de empresa		Dígito de extensión y referencia de la serie	
	Bits *(M)*	**Dígitos *(L)***	**Bits *(N)***	**Dígitos**
0	40	12	18	5
1	37	11	21	6
2	34	10	24	7
3	30	9	28	8
4	27	8	31	9
5	24	7	34	10
6	20	6	38	11

14.5.2.1 Tabla de codificación SSCC-96

Tabla 14-6 Tabla de codificación SSCC-96

Esquema	SSCC-96					
Plantilla de URI	urn:epc:tag:sscc-96:*F.C.S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Extensión / referencia de la serie	(Reservado)
Cómputo de bits del segmento lógico	8	3	3	20-40	38-18	24
Segmento de codificación	Encabezado de EPC	Filtro	SSCC			(Reservado)
Porción del URI		*F*	*C.S*			
Cómputo de bits del segmento de codificación	8	3	61			24
Posición binaria	***b***95***b***94…***b***88	***b***87***b***86***b***85	***b***84***b***83…***b***24			***b***23***b***36…***b***0
Método de codificación	00110001	Entero	Partición *Tabla 14-5*			00…0 (24 bits cero)

14.5.3 Número de localización global con o sin extensión (SGLN)

Se especifican dos esquemas de codificación para el SGLN, una codificación de 96 bits (SGLN-96) y una codificación de 195 bits (SGLN-195). La codificación SGLN-195 permite el rango completo de extensiones de GLN hasta por 20 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación SGLN-96 permite extensiones de GLN únicamente numéricas, sin ceros iniciales, cuyo valor es menor que 2^{41} (es decir, de 0 a 2,199,023,255,551, inclusive). Tome en cuenta que un valor de extensión de 0 se reserva para indicar que el SGLN equivale al GLN indicado por el Prefijo GS1 de empresa y la referencia de la localización; este valor está disponible en las codificaciones SGLN-96 y SGLN-195.

Ambos esquemas de codificación de SGLN hacen referencia a la tabla de partición siguiente.

Tabla 14-7 Tabla de partición de SGLN

Valor de la partición *(P)*	Prefijo GS1 de empresa		Referencia de la localización	
	Bits (*M*)	**Dígitos (*T*)**	**Bits (*N*)**	**Dígitos**
0	40	12	1	0
1	37	11	4	1
2	34	10	7	2
3	30	9	11	3
4	27	8	14	4
5	24	7	17	5
6	20	6	21	6

14.5.3.1 Tabla de codificación SGLN-96

Tabla 14-8 Tabla de codificación SGLN-96

Esquema	SGLN-96					
Plantilla de URI	urn:epc:tag:sgln-96:*F.C.L.E*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	GS1 empresa Prefijo	Localización Referencia	Extensión
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	41
Segmento de codificación	Encabezado de EPC	Filtro	GLN			Extensión
Porción del URI		*F*	*C.L*			*E*
Cómputo de bits del segmento de codificación	8	3	44			41
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{41}$			$b_{40}b_{39}\ldots b_{0}$
Método de codificación	00110010	Entero	Partición *Tabla 14-7*			Entero

14.5.3.2 Tabla de codificación SGLN-195

Tabla 14-9 Tabla de codificación SGLN-195

Esquema	SGLN-195					
Plantilla de URI	urn:epc:tag:sgln-195:*F.C.L.E*					
Total de bits	195					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia de la localización	Extensión
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	140
Segmento de codificación	Encabezado de EPC	Filtro	GLN			Extensión

Esquema	SGLN-195			
Porción del URI		*F*	*C.L*	*E*
Cómputo de bits del segmento de codificación	8	3	44	140
Posición binaria	$b_{194}b_{193}...b_{187}$	$b_{186}b_{185}b_{184}$	$b_{183}b_{182}...b_{140}$	$b_{139}b_{138}...b_0$
Método de codificación	00111001	Entero	Partición *Tabla 14-7*	Secuencia

14.5.4 Identificador global de activos retornables (GRAI)

Se especifican dos esquemas de codificación para el GRAI, una codificación de 96 bits (GRAI-96) y una codificación de 170 bits (GRAI-170). La codificación GRAI-170 permite el rango completo de números de serie hasta por 16 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación GRAI-96 permite los números de serie únicamente numéricos, sin ceros iniciales, cuyo valor es menor que 2^{38} (es decir, de 0 a 274,877,906,943 inclusive).

Solo los GRAI que incluyen el número de serie opcional se pueden representar como EPC. Un GRAI sin un número de serie representa una clase de activo, en lugar de una instancia específica, y por lo tanto no se puede usar como EPC (tal como un GTIN no seriado no se puede utilizar como EPC).

Ambos esquemas de codificación de GRAI hacen referencia a la tabla de partición siguiente.

Tabla 14-10 Tabla de partición de GRAI

Valor de la partición *(P)*	Prefijo de empresa		Tipo de activo	
	Bits *(M)*	**Dígitos *(L)***	**Bits (N)**	**Dígitos**
0	40	12	4	0
1	37	11	7	1
2	34	10	10	2
3	30	9	14	3
4	27	8	17	4
5	24	7	20	5
6	20	6	24	6

14.5.4.1 Tabla de codificación GRAI-96

Tabla 14-11 Tabla de codificación GRAI-96

Esquema	GRAI-96					
Plantilla de URI	urn:epc:tag:grai-96:*F.C.A.S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Tipo de activo	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	38
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa+ Tipo de activo			Seriado
Porción del URI		*F*	*C.A*			*S*

Esquema	GRAI-96			
Codificación Segmento Cómputo de bits	8	3	47	38
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{38}$	$b_{37}b_{36}\ldots b_{0}$
Método de codificación	00110011	Entero	Partición *Tabla 14-10*	Entero

14.5.4.2 Tabla de codificación GRAI-170

Tabla 14-12 Tabla de codificación GRAI-170

Esquema	GRAI-170					
Plantilla de URI	`urn:epc:tag:grai-170:`*F.C.A.S*					
Total de bits	170					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Tipo de activo	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	112
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa+ Tipo de activo			Seriado
Porción del URI		*F*	*C.A*			*S*
Cómputo de bits del segmento de codificación	8	3	47			112
Posición binaria	$b_{169}b_{168}\ldots b_{162}$	$b_{161}b_{160}b_{159}$	$b_{158}b_{157}\ldots b_{112}$			$b_{111}b_{110}\ldots b_{0}$
Método de codificación	00110111	Entero	Partición *Tabla 14-10*			Secuencia

14.5.5 Identificador global individual de activo (GIAI)

Se especifican dos esquemas de codificación para el GIAI, una codificación de 96 bits (GIAI-96) y una codificación de 202 bits (GIAI-202). La codificación GIAI-202 permite el rango completo de números de serie hasta por 24 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación GIAI-96 permite números de serie únicamente numéricos, sin ceros iniciales, cuyo valor alcanza un límite que varía con la longitud del Prefijo GS1 de empresa.

Cada esquema de codificación de GIAI hace referencia a una tabla de partición diferente, especificada junto con la tabla de codificación correspondiente en las subsecciones siguientes.

14.5.5.1 Tabla de partición y Tabla de codificación GIAI-96

El esquema de codificación GIAI-96 hace uso de la tabla de partición siguiente.

Tabla 14-13 Tabla de partición GIAI-96

Valor de la partición *(P)*	Prefijo de empresa		Referencia individual del activo	
	Bits *(M)*	Dígitos (*L*)	Bits *(N)*	Cant. máxima de dígitos (K)
0	40	12	42	13
1	37	11	45	14

Valor de la partición (*P*)	Prefijo de empresa		Referencia individual del activo	
2	34	10	48	15
3	30	8	52	16
4	27	8	55	17
5	24	7	58	18
6	20	6	62	19

Tabla 14-14 Tabla de codificación GIAI-96

Esquema	GIAI-96				
Plantilla de URI	`urn:epc:tag:giai-96:`*`F.C.A`*				
Total de bits	96				
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia individual del activo
Cómputo de bits del segmento lógico	8	3	3	20-40	62-42
Segmento de codificación	Encabezado de EPC	Filtro	GIAI		
Porción del URI		*F*	*C.A*		
Cómputo de bits del segmento de codificación	8	3	85		
Posición binaria	$b_{95}b_{94}...b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}...b_{0}$		
Método de codificación	00110100	Entero	Partición sin relleno *Tabla 14-13 Tabla 14-14*		

14.5.5.2 Tabla de partición y Tabla de codificación GIAI-202

El esquema de codificación GIAI-202 hace uso de la tabla de partición siguiente.

Tabla 14-15 Tabla de partición GIAI-202

Valor de la partición (*P*)	Prefijo de empresa		Referencia individual del activo	
	Bits (*M*)	**Dígitos (*L*)**	**Bits (*N*)**	**Cantidad máxima de caracteres**
0	40	12	148	18
1	37	11	151	19
2	34	10	154	20
3	30	9	158	21
4	27	8	161	22
5	24	7	164	23
6	20	6	168	24

Tabla 14-16 Tabla de codificación GIAI-202

Esquema	GIAI-202
Plantilla de URI	`urn:epc:tag:giai-202:`*`F.C.A`*
Total de bits	202

Esquema	GIAI-202				
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia individual del activo
Cómputo de bits del segmento lógico	8	3	3	20-40	168-148
Segmento de codificación	Encabezado de EPC	Filtro	GIAI		
Porción del URI		*F*	*C*.*A*		
Cómputo de bits del segmento de codificación	8	3	191		
Posición binaria	$b_{201}b_{200}\ldots b_{194}$	$b_{193}b_{192}b_{191}$	$b_{190}b_{189}\ldots b_{0}$		
Método de codificación	00111000	Entero	Partición de la cadena *Tabla 14-15*		

14.5.6 Número global de relación del servicio (GSRN)

Se especifican dos esquemas de codificación para el GSRN: la codificación GSRN-96 de 96 bits y la codificación GSRNP-96 de 96 bits. Ambas codificaciones GSRN-96 permiten el rango completo de códigos GSRN, según se especifica en [GS1GS].

Ambos esquemas de codificación de GSRN-96 hacen referencia a la tabla de partición siguiente.

Tabla 14-17 Tabla de partición de GSRN

Valor de la partición (***P***)	Prefijo de empresa		Referencia de servicio	
	Bits (*M*)	**Dígitos (*L*)**	**Bits (*N*)**	**Dígitos**
0	40	12	18	5
1	37	11	21	6
2	34	10	24	7
3	30	9	28	8
4	27	8	31	9
5	24	7	34	10
6	20	6	38	11

14.5.6.1 Tabla de codificación GSRN-96

Tabla 14-18 Tabla de codificación GSRN-96

Esquema	GSRN-96					
Plantilla de URI	`urn:epc:tag:gsrn-96:`*F*.*C*.*S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia de servicio	(Reservado)
Cómputo de bits del segmento lógico	8	3	3	20-40	38-18	24
Segmento de codificación	Encabezado de EPC	Filtro	GSRN			(Reservado)

Esquema	GSRN-96			
Porción del URI		*F*	*C*.*S*	
Cómputo de bits del segmento de codificación	8	3	61	24
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{24}$	$b_{23}b_{22}\ldots b_{0}$
Método de codificación	00101101	Entero	Partición *Tabla 14-17*	00.0 (24 bits cero)

14.5.6.2 Tabla de codificación GSRNP-96

Tabla 14-19 Tabla de codificación GSRNP-96

Esquema	GSRNP-96					
Plantilla de URI	urn:epc:tag:gsrnp-96:*F*.*C*.*S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia de servicio	(Reservado)
Cómputo de bits del segmento lógico	8	3	3	20-40	38-18	24
Segmento de codificación	Encabezado de EPC	Filtro	GSRN			(Reservado)
Porción del URI		*F*	*C*.*S*			
Cómputo de bits del segmento de codificación	8	3	61			24
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{24}$			$b_{23}b_{22}\ldots b_{0}$
Método de codificación	00101110	Entero	Partición *Tabla 14-17*			00.0 (24 bits cero)

14.5.7 Identificador de tipo de documento global (GDTI)

Se especifican tres esquemas de codificación para el GDTI, una codificación de 96 bits (GDTI-96), una codificación de 113 bits (GDTI-113, OBSOLETA desde TDS 1.9) y una codificación de 174 bits (GDTI-174). La codificación GDTI-174 permite el rango completo de serialización de documentos hasta por 17 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación GDTI-113 obsoleta permite un rango reducido de números de serie de documento hasta por 17 caracteres numéricos (incluidos los ceros iniciales), según se especifica originalmente en [GS1GS11.0]. La codificación GDTI-96 permite los números de serie de documento, sin ceros iniciales, cuyo valor es menor que 2^{41} (es decir, de 0 a 2,199,023,255,551, inclusive).

Solo los GDTI que incluyen el número de serie opcional se pueden representar como EPC. Un GDTI sin un número de serie representa una clase de documento, en lugar de un documento específico, y por lo tanto no se puede usar como EPC (tal como un GTIN no seriado no se puede utilizar como EPC).

Ambos esquemas de codificación de GDTI hacen referencia a la tabla de partición siguiente.

Tabla 14-20 Tabla de partición de GDTI

Valor de la partición ***(P)***	Prefijo de empresa		Tipo de documento	
	Bits *(M)*	**Dígitos *(L)***	**Bits *(N)***	**Dígitos**
0	40	12	1	0

Valor de la partición (*P*)	Prefijo de empresa		Tipo de documento	
1	37	11	4	1
2	34	10	7	2
3	30	9	11	3
4	27	8	14	4
5	24	7	17	5
6	20	6	21	6

14.5.7.1 Tabla de codificación GDTI-96

Tabla 14-21 Tabla de codificación GDTI-96

Esquema	GDTI-96					
Plantilla de URI	`urn:epc:tag:gdti-96:`*`F.C.D.S`*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Tipo de documento	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	41
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa+ Tipo de documento			Seriado
Porción del URI		*F*	*C.D*			*S*
Cómputo de bits del segmento de codificación	8	3	44			41
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{41}$			$b_{40}b_{39}\ldots b_{0}$
Método de codificación	00101100	Entero	Partición *Tabla 14-20*			Entero

14.5.7.2 Tabla de codificación GDTI-113

Tabla 14-22 Tabla de codificación GDTI-113

Esquema	GDTI-113					
Plantilla de URI	`urn:epc:tag:gdti-113:`*`F.C.D.S`*					
Total de bits	113					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Tipo de documento	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	58
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa+ Tipo de documento			Seriado
Porción del URI		*F*	*C.D*			*S*

Esquema	GDTI-113			
Cómputo de bits del segmento de codificación	8	3	44	58
Posición binaria	$b_{112}b_{111}\ldots b_{105}$	$b_{104}b_{103}b_{102}$	$b_{101}b_{100}\ldots b_{58}$	$b_{57}b_{56}\ldots b_{c}$
Método de codificación	00111010	Entero	Partición *Tabla 14-20*	Cadena numérica

14.5.7.3 Tabla de codificación GDTI-174

Tabla 14-23 Tabla de codificación GDTI-174

Esquema	GDTI-174					
Plantilla de URI	`urn:epc:tag:gdti-174:`*F.C.A.S*					
Total de bits	174					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Tipo de documento	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	119
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa + Activo Tipo			Seriado
Porción del URI		*F*	*C.A*			*S*
Cómputo de bits del segmento de codificación	8	3	44			119
Posición binaria	$b_{173}b_{172}\ldots b_{166}$	$b_{165}b_{164}b_{163}$	$b_{162}b_{161}\ldots b_{119}$			$b_{118}b_{117}\ldots b_{0}$
Método de codificación	00111110	Entero	Partición *Tabla 14-20*			Secuencia

14.5.8 Identificador de CPI (CPI)

Se especifican dos esquemas de codificación para el identificador CPI: el esquema CPI-96 de 96 bits y la codificación CPI-var de longitud variable. CPI-96 utiliza la tabla de partición 39 y CPI-var utiliza la tabla de partición 40.

Tabla 14-24 Tabla de partición CPI-96

Valor de la partición (***P***)	Prefijo GS1 de empresa		Referencia del componente/pieza	
	Bits (*M*)	**Dígitos (*L*)**	**Bits (N)**	**Cantidad máxima de dígitos**
0	40	12	11	3
1	37	11	14	4
2	34	10	17	5
3	30	9	21	6
4	27	8	24	7
5	24	7	27	8
6	20	6	31	9

Tabla 14-25 Tabla de partición de CPI-var

Valor de la partición (*P*)	Prefijo GS1 de empresa		Referencia del componente/pieza	
	Bits (*M*)	**Dígitos (*L*)**	**Cantidad máxima de bits ** (N)**	**Cantidad máxima de caracteres**
0	40	12	114	18
1	37	11	120	19
2	34	10	126	20
3	30	9	132	21
4	27	8	138	22
5	24	7	144	23
6	20	6	150	24

** La cantidad de bits depende de la cantidad de caracteres en la referencia del componente/pieza; vea las secciones *14.3.9* y *14.4.9*.

14.5.8.1 Tabla de codificación CPI-96

Tabla 14-26 Tabla de codificación CPI-96

Esquema	CPI-96					
Plantilla de URI	urn:epc:tag:cpi-96:*F.C.P.S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia del componente/pieza	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	31-11	31
Segmento de codificación	Encabezado de EPC	Filtro	Identificador de componente/pieza			Número de serie del componente/pieza
Porción del URI		*F*	*C.P*			*S*
Cómputo de bits del segmento de codificación	8	3	54			31
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{31}$			$b_{30}b_{29}\ldots b_{0}$
Método de codificación	00111100	Entero	Partición sin relleno *Tabla 14-24*			Entero

14.5.8.2 Tabla de codificación CPI-var

Tabla 14-27 Tabla de codificación CPI-var

Esquema	CPI-var					
Plantilla de URI	urn:epc:tag:cpi-var:*F.C.P.S*					
Total de bits	Variable: entre 86 y 224 bits (inclusive)					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia de componente/pieza	Seriado

Esquema	CPI-var					
Cómputo de bits del segmento lógico	8	3	3	20-40	12-150 (variable)	40 (fijo)
Segmento de codificación	Encabezado de EPC	Filtro	Identificador de componente/pieza			Número de serie de componente/pieza
Porción del URI		*F*	*C.P*			*S*
Cómputo de bits del segmento de codificación	8	3	Hasta 173 bits			40
Posición binaria	$b_{B-1}b_{B-2}\ldots b_{B-8}$	$b_{B-9}b_{B-10}b_{B-11}$	$b_{B-12}b_{B-13}\ldots b_{40}$			$b_{39}b_{38}\ldots b_0$
Método de codificación	00111101	Entero	Partición de la cadena de longitud variable de 6 bits *Tabla 14-25*			Entero

14.5.9 Número de cupón global (SGCN)

Se especifica un esquema de codificación único de 96 bits (SGCN-96) para el SGCN, lo que permite el rango completo de números de serie de componente de cupón hasta por 12 caracteres numéricos (incluidos los ceros iniciales), según se especifica en [GS1GS]. Solo los SGCN que incluyen el número de serie se pueden representar como EPC. Un GCN sin un número de serie representa una clase de cupón, en lugar de un cupón específico, y por lo tanto no se puede usar como EPC (tal como un GTIN no seriado no se puede utilizar como EPC).

El esquema de codificación SGCN hace referencia a la tabla de partición siguiente.

Tabla 14-28 Tabla de partición de SGCN

Valor de la partición (*P*)	Prefijo de empresa		Referencia del cupón	
	Bits (*M*)	**Dígitos (*L*)**	**Bits (*N*)**	**Dígitos**
0	40	12	1	0
1	37	11	4	1
2	34	10	7	2
3	30	9	11	3
4	27	8	14	4
5	24	7	17	5
6	20	6	21	6

14.5.9.1 Tabla de codificación SGCN-96

Tabla 14-29 Tabla de codificación SGCN-96

Esquema	SGCN-96					
Plantilla de URI	`urn:epc:tag:sgcn-96:`*F.C.D.S*					
Total de bits	96					
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa	Referencia del cupón	Componente serial
Cómputo de bits del segmento lógico	8	3	3	20-40	21-1	41

Esquema	SGCN-96			
Segmento de codificación	Encabezado de EPC	Filtro	Partición + Prefijo de empresa + Referencia del cupón	Seriado
Porción del URI		*F*	*C.D*	*S*
Cómputo de bits del segmento de codificación	8	3	44	41
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}b_{85}$	$b_{84}b_{83}\ldots b_{41}$	$b_{40}b_{39}\ldots b_{0}$
Método de codificación	00111111	Entero	Partición *Tabla 14-28*	Cadena numérica

14.5.10 Artículo comercial individual (ITIP)

Se especifican dos esquemas de codificación para el ITIP, una codificación de 110 bits (ITIP-110) y una codificación de 212 bits (ITIP-212). La codificación ITIP-212 permite el rango completo de números de serie hasta por 20 caracteres alfanuméricos, según se especifica en [GS1GS]. La codificación ITIP-110 permite los números de serie únicamente numéricos, sin ceros iniciales, cuyo valor es menor que 2^{38} (es decir, de 0 a 274,877,906,943 inclusive).

Ambos esquemas de codificación de ITIP hacen referencia a la tabla de partición siguiente.

Tabla 14-30 Tabla de partición de ITIP

Valor de la partición (*P*)	Prefijo GS1 de empresa		Indicador/dígito de relleno y referencia del artículo	
	Bits (*M*)	**Dígitos (*L*)**	**Bits (N)**	**Dígitos**
0	40	12	4	1
1	37	11	7	2
2	34	10	10	3
3	30	9	14	4
4	27	8	17	5
5	24	7	20	6
6	20	6	24	7

14.5.10.1 Tabla de codificación ITIP-110

Tabla 14-31 Tabla de codificación ITIP-110

Esquema	ITIP-110							
Plantilla de URI	urn:epc:tag:itip-110:*F. C. I.PT. S*							
Total de bits	110							
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa (*)	Indicador (**) / Referencia del artículo	Pieza	Total	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	7	7	38
Segmento de codificación	Encabezado de EPC	Filtro	GTIN			Pieza	Total	Seriado
Porción del URI		*F*	*C.I*			*P*	*T*	*S*

Esquema	ITIP-110					
Cómputo de bits del segmento de codificación	8	3	47	7	7	38
Posición binaria	$b_{109}b_{108}...b_{102}$	$b_{101}b_{100}b_{99}$	$b_{98}b_{97}...b_{52}$	$b_{51}b_{50}...b_{45}$	$b_{44}b_{43}...b_{38}$	$b_{37}b_{36}...b_{0}$
Método de codificación	01000000	Entero	Partición *Tabla 14-2*	Entero de ancho fijo	Entero de ancho fijo	Entero

(*) Vea la Sección *7.1.2* para conocer el caso de un SGTIN derivado de GTIN-8.

(**) Tome en cuenta que en el caso de un ITIP derivado de un GTIN-12 o GTIN-13, un dígito cero de relleno toma el lugar del dígito indicador. En todos los casos, vea la Sección *7.1* para conocer la definición de la manera en que el dígito indicado (o dígito cero de relleno) y la referencia del artículo se combinan en este segmento del EPC.

14.5.10.2 Tabla de codificación ITIP-212

Tabla 14-32 Tabla de codificación ITIP-212

Esquema	ITIP-212							
Plantilla de URI	`urn:epc:tag:itip-212:`*`F. C. I.PT. S`*							
Total de bits	212							
Segmento lógico	Encabezado de EPC	Filtro	Partición	Prefijo GS1 de empresa (*)	Indicador (**) / Referencia del artículo	Pieza	Total	Seriado
Cómputo de bits del segmento lógico	8	3	3	20-40	24-4	7	7	140
Segmento de codificación	Encabezado de EPC	Filtro	GTIN			Pieza	Total	Seriado
Porción del URI		*F*	*C.I*			*P*	*T*	*S*
Cómputo de bits del segmento de codificación	8	3	47			7	7	140
Posición binaria	$b_{211}b_{210}...b_{204}$	$b_{203}b_{202}b_{201}$	$b_{200}b_{199}...b_{154}$			$b_{153}b_{152}...b_{147}$	$b_{146}b_{145}...b_{140}$	$b_{139}b_{138}...b_{0}$
Método de codificación	01000001	Entero	Partición *Tabla 14-2*			Entero de ancho fijo	Entero de ancho fijo	Secuencia

(*) Vea la Sección *7.1.2* para conocer el caso de un SGTIN derivado de GTIN-8.

(**) Tome en cuenta que en el caso de un ITIP derivado de un GTIN-12 o GTIN-13, un dígito cero de relleno toma el lugar del dígito indicador. En todos los casos, vea la Sección *7.1* para conocer la definición de la manera en que el dígito indicado (o dígito cero de relleno) y la referencia del artículo se combinan en este segmento del EPC.

14.5.11 Identificador general (GID)

Se especifica un esquema de codificación para el GID: el GID-96 de codificación de 96 bits. No se requiere una tabla de partición.

14.5.11.1 Tabla de codificación GID-96

Tabla 14-33 Tabla de codificación GID-96

Esquema	GID-96			
Plantilla de URI	urn:epc:tag:gid-96:*M. C.S*			
Total de bits	96			
Segmento lógico	Encabezado de EPC	Número de gerente general	Clase de objeto	Número de serie
Cómputo de bits del segmento lógico	8	28	24	36
Segmento de codificación	Encabezado de EPC	Número de gerente general	Clase de objeto	Número de serie
Porción del URI		*M*	*C*	*S*
Cómputo de bits del segmento de codificación	8	28	24	36
Posición binaria	$b_{95}b_{94}\ldots b_{88}$	$b_{87}b_{86}\ldots b_{60}$	$b_{59}b_{58}\ldots b_{36}$	$b_{35}b_{34}\ldots b_{0}$
Método de codificación	00110101	Entero	Entero	Entero

14.5.12 Identificador DoD

En el momento de la redacción de este texto, los detalles de la codificación de DoD aparecen en un documento titulado "United States Department of Defense Supplier's Passive RFID Information Guide", que se puede obtener en el sitio web del Departamento de Defensa de los Estados Unidos (http://www.dodrfid.org/supplierguide.htm).

14.5.13 Identificador ADI (ADI)

Se especifica un esquema de codificación para el identificador ADI: la codificación de longitud variable ADI-var. No se requiere una tabla de partición.

14.5.13.1 Tabla de codificación ADI-var

Tabla 14-34 Tabla de codificación ADI-var

Esquema	ADI-var				
Plantilla de URI	urn:epc:tag:adi-var:*F.D.P.S*				
Total de bits	Variable: entre 68 y 434 bits (inclusive)				
Segmento lógico	Encabezado de EPC	Filtro	CAGE/ DoDAAC	Número de parte	Número de serie
Cómputo de bits del segmento lógico	8	6	36	Variable	Variable
Codificación Segmento	Encabezado de EPC	Filtro	CAGE/ DoDAAC	Número de parte	Número de serie
Porción del URI		*F*	*D*	*P*	*S*
Cómputo de bits del segmento de codificación	8	6	36	Variable (6 - 198)	Variable (12 - 186)
Posición binaria	$b_{B-1}b_{B-2}.b_{B-8}$	$b_{B-9}b_{B-10}..b_{B-14}$	$b_{B-15}b_{B-16}..b_{B-50}$	$b_{B-51}b_{B-52}\ldots$	$\ldots b1b_0$
Método de codificación	00111011	Entero	CAGE/ DoDAAC de 6 bits	Cadena variable de 6 bits	Cadena variable de 6 bits

Notas:

La cantidad de caracteres en el segmento del número de parte debe ser mayor o igual que cero y menor o igual que 32. En la codificación binaria, siempre está presente el carácter de terminación cero de 6 bits.

La cantidad de caracteres en el segmento del número de parte debe ser mayor o igual que uno y menor o igual que 30. En la codificación binaria, siempre está presente el carácter de terminación cero de 6 bits.

El carácter "#" (representado en el URI por la secuencia %23 de escape) puede aparecer como el primer carácter del segmento de número de serie, pero por lo demás es posible que no aparezca en el segmento del número de parte ni en ninguna otra parte del segmento del número de serie.

15 Contenido del banco de memoria de EPC

En esta sección se especifica cómo se traduce el URI de etiqueta de EPC y el URI de datos sin procesar de EPC en el contenido binario del banco de memoria de EPC de una etiqueta Gen 2, y viceversa.

15.1 Procedimientos de codificación

En esta sección se especifica cómo se traduce el URI de etiqueta de EPC y el URI de datos sin procesar de EPC en el contenido binario del banco de memoria de EPC de una etiqueta Gen 2.

15.1.1 URI de etiqueta de EPC en la memoria de memoria de EPC Gen 2

Datos:

- Un URI de etiqueta de EPC que comienza con `urn:epc:tag`:

Procedimiento de codificación:

1. Si el URI no es válido sintácticamente de acuerdo con la Sección *12.4*, deténgase: este URI no se puede codificar.
2. Aplique el procedimiento de codificación de la Sección *14.3* al URI. El resultado es una cadena binaria de *N* bits. Si la codificación de codificación falla, deténgase: este URI no se puede codificar.
3. Llene el banco de memoria de EPC Gen 2 de acuerdo con la tabla siguiente:

Tabla 15-1 Indicaciones para llenar el banco de memoria de EPC Gen 2 con el URI de etiqueta de EPC

Bits	Campo	Contenido
00_h - $0F_h$	CRC	Código de CRC calculado a partir del resto del banco de memoria. (Por lo general, se calcula automáticamente mediante el lector, por lo que el software que implementa este procedimiento no tiene que encargarse de ello).
10_h - 14_h	Longitud	La cantidad de bits, *N*, en la codificación binaria de EPC determinada en el Paso 2 anterior, dividida entre 16, y redondeada al siguiente entero más alto si *N* no es un múltiplo de 16.
15_h	Indicador de memoria del usuario	Si el URI de etiqueta de EPC incluye un campo de control `[umi=1]`, un bit de uno. Si el URI de etiqueta de EPC incluye un campo de control `[umi=0]` o no contiene un campo de control umi, un bit de cero. Tome en cuenta que es posible que algunas etiquetas de generación 2 ignoren el valor escrito para este bit y, en su lugar, calculen el valor del bit a partir del contenido de la memoria del usuario. Consulte [UHFC1G2].
16_h	Indicador de XPC	Este bit se calcula con la etiqueta y se ignora en la etiqueta cuando esta contiene texto; por lo tanto, no se considera en este procedimiento de codificación.
17_h	Conmutación	0, que indica el banco de EPC que contiene un EPC
18_h - $1F_h$	Bits de atributo	Si el URI de etiqueta de EPC Tag incluye un campo de control `[att=xNN]`, el valor NN considerado como un número hexadecimal de 8 bits. Si el URI de etiqueta de EPC no contiene ese campo de control de cero.
20_h - ?	EPC / UII	Los bits *N* obtenidos a partir del procedimiento de codificación binaria de EPC en el paso 2 anterior, seguido de los ceros suficientes como para que la cantidad total de bits sea un múltiplo de 16 (0 - 15 bits adicionales de cero)

Reglas no normativas: Explicación (no normativa): Los bits de XPC (bits 210_h - $21F_h$) no están incluidos en este procedimiento, porque los únicos bits de XPC definidos en [UHFC1G2] son bits escritos indirectamente mediante repetición de puesta en servicio. Esos bits no se deben escribir explícitamente mediante una aplicación.

15.1.2 URI de datos sin procesar de EPC en la memoria de memoria de EPC Gen 2

Datos:

- Un URI de datos in procesar de EPC que comienza con urn:epc:raw:. Ese tipo de URI tiene una de las tres formas siguientes:

  `urn:epc:raw:`*`OptionalControlFields`*`:`*`Length`*`.x`*`HexPayload`*

  `urn:epc:raw:`*`OptionalControlFields`*`:`*`Length`*`.x`*`AFI`*`.x`*`HexPayload`*

  `urn:epc:raw:`*`OptionalControlFields`*`:`*`Length`*`.`*`DecimalPayload`*

Procedimiento de codificación:

1. Si el URI no es válido sintácticamente de acuerdo con la sintaxis en la Sección 12.4, deténgase: este URI no se puede codificar.
2. Extraiga el elemento `NonZeroComponent` de la izquierda de acuerdo con la sintaxis (el campo *Longitud* en las plantillas anteriores). Este componente aparece inmediatamente después del carácter de dos puntos (:) de la derecha. Considere esto como un entero decimal, *N*. Este es la cantidad de bits en la carga útil que no contiene datos sin procesar.
3. Determine el bit de alternar y el AFI (si lo hay):
   a. Si la parte principal del URI coincide con la producción de `DecimalRawURIBody` o de `HexRawURIBody` de la sintaxis (la primera y tercera plantillas anteriores), el bit de alternar es igual a cero.
   b. Si la parte principal del URI coincide con la producción de `AFIRawURIBody` de la sintaxis (la segunda plantilla anterior), el bit de alternar es uno. El AFI es el valor del elemento `HexComponent` de la izquierda en `AFIRawURIBody` (el campo *AFI* en la plantilla anterior), considerado como un entero hexadecimal sin signo de 8 bits. Si el valor de `HexComponent` es mayor o igual que 256, deténgase: este URI no se puede codificar.
4. Determine la carga útil de EPC/UII:
   c. Si el cuerpo del URI coincide con la producción de `HexRawURIBody` de la sintaxis (primera plantilla anterior) o con la producción de `AFIRawURIBody` de la sintaxis (segunda plantilla anterior), la carga útil es el elemento `HexComponent` de la derecha en el cuerpo (el campo *HexPayload* en las plantillas anteriores), considerado como un entero hexadecimal sin signos con bits *N*, donde *N* se determina según el paso 2 anterior. Si el valor de HexComponent es mayor o igual que 2^N, deténgase: este URI no se puede codificar.
   d. Si el cuerpo del URI coincide con la producción de `DecimalRawURIBody` de la sintaxis (tercera plantilla anterior), la carga útil es el elemento `NumericComponent` de la derecha en el cuerpo (el campo `DecimalPayload` en la plantilla anterior), considerado como un entero decimal sin signos con bits *N*, donde *N* se determina según el paso 2 de arriba. Si el valor de este elemento `NumericComponent` es mayor o igual que 2^N, deténgase: este URI no se puede codificar.
5. Llene el banco de memoria de EPC Gen 2 de acuerdo con la tabla siguiente:

Tabla 15-2 Indicaciones para llenar el banco de memoria de EPC Gen 2 con el URI de datos sin procesar de EPC

Bits	Campo	Contenido
00_h - $0F_h$	CRC	Código de CRC calculado a partir del resto del banco de memoria. (Por lo general, se calcula automáticamente mediante el lector, por lo que el software que implementa este procedimiento no tiene que encargarse de ello).
10_h - 14_h	Longitud	La cantidad de bits, *N*, en la codificación binaria de EPC determinada en el Paso 2 anterior, dividida entre 16, y redondeada al siguiente entero más alto si *N* no es un múltiplo de 16.

Bits	Campo	Contenido
15_h	Indicador de memoria del usuario	Este bit se calcula con la etiqueta y se ignora en la etiqueta cuando esta contiene texto; por lo tanto, no se considera en este procedimiento de codificación.
16_h	Indicador de XPC	Este bit se calcula con la etiqueta y se ignora en la etiqueta cuando esta contiene texto; por lo tanto, no se considera en este procedimiento de codificación.
17_h	Conmutación	El valor determinado en el paso 3 anterior.
18_h - $1F_h$	AFI / Bits de atributo	Si el valor de alternar determinado en el paso 3 es uno, el valor del AFI se determina en el paso 3.2. De lo contrario, Si el URI incluye un campo de control [`att=xNN`], el valor NN considerado como un número hexadecimal de 8 bits. Si el URI no contiene ese campo de control, es cero.
20_h - ?	EPC / UII	Los bits *N* determinados en el paso 4 arriba, seguido de los ceros suficientes como para que la cantidad total de bits sea un múltiplo de 16 (0 - 15 bits adicionales de cero)

15.2 Procedimientos de decodificación

En esta sección se especifica cómo traducir el contenido binario del banco de memoria de EPC de una etiqueta Gen 2 en el URI de etiqueta de EPC y URI de datos sin procesar de EPC.

15.2.1 Banco de memoria de EPC Gen 2 en URI de datos sin procesar de EPC

Datos:

- Contenido del banco de memoria de EPC de una etiqueta Gen 2

Procedimiento:

1. Extraiga los bits de longitud, bits 10_h - 14_h. Considere estos bits como un entero sin signo *L*.
2. Calcule *N* = 16*L*.
3. Si bit 17_h se establece en uno, extraiga los bits 18_h - $1F_h$ y considérelos como un entero si signo *A*. Construya una cadena que conste de una letra “x”, seguida de *A* como un numeral hexadecimal de 2 dígitos (que usa dígitos y mayúsculas únicamente), seguido de un punto (“.”).
4. Aplique el procedimiento de decodificación de la Sección *15.2.4* para decodificar los campos de control.
5. Extraiga bits *N* que comiencen con el bit 20_h y considérelos como un entero sin signo *V*. Construya una cadena que conste de la letra “x” seguida de *V* como un numeral hexadecimal de (*N*/4) dígitos (con dígitos y mayúsculas únicamente).
6. Construya una cadena que conste de “`urn:epc:raw:`”, seguida del resultado del paso 4 (si no está vacío), seguido de *N* como un numeral decimal sin ceros iniciales, seguido de un punto (“`.`”), seguido del resultado del paso 3 (si no está vacío), seguido del resultado del paso 5. Este es el URI final de datos sin procesar de EPC.

15.2.2 Banco de memoria de EPC Gen 2 en URI de etiqueta de EPC

Este procedimiento decodifica el contenido de un banco de memoria de EPC Gen 2 en un URI de etiqueta de EPC comenzando con `urn:epc:tag:` si la memoria contiene un EPC válido, o un URI con datos sin procesar de EPC que comience con `urn:epc:raw:`.

Datos:

- Contenido del banco de memoria de EPC de una etiqueta Gen 2

Procedimiento:

1. Extraiga los bits de longitud, bits 10h - 14h. Considere estos bits como un entero sin signo *L*.
2. Calcule *N* = 16*L*.

3. Extraiga los bits *N* comenzando con el bit 20h. Aplique el procedimiento de decodificación de la Sección 14.3.9, usando los bits *N* como la entrada del procedimiento.

4. Si el procedimiento de decodificación de la Sección 14.3.9 falla, continúe el procedimiento de decodificación de la Sección 15.2.1 para calcular un URI de datos sin procesar de EPC. De otro modo, el procedimiento de decodificación de la Sección 14.3.9 arroja un URI de etiqueta de EPC que comienza con `urn:epc:tag:`. Continúe con el paso siguiente.

5. Aplique el procedimiento de decodificación de la Sección 15.2.4 para decodificar los campos de control.

6. Inserte el resultado de la Sección 15.2.4 (incluidos dos puntos iniciales) en el URI de etiqueta de EPC obtenido en el paso 4, inmediatamente después del prefijo `urn:epc:tag:`. (Si la Sección 15.2.4 tiene como resultado una cadena vacía, este resultado es idéntico al que se obtiene en el paso 4.) El resultado es el URI final de etiqueta de EPC.

15.2.3 Banco de memoria de EPC Gen 2 en un URI de EPC de identidad pura

Este procedimiento decodifica el contenido de un banco de memoria de EPC Gen 2 en un URI de etiqueta de EPC comenzando con `urn:epc:tag:` si la memoria contiene un EPC válido, o un URI con datos sin procesar de EPC que comience con `urn:epc:raw:`.

Datos:

- Contenido del banco de memoria de EPC de una etiqueta Gen 2

Procedimiento:

1. Aplique el procedimiento de decodificación de la Sección 15.2.2 para obtener un URI de etiqueta de EPC o un URI de datos sin procesar de EPC. Si se obtiene un URI con datos sin procesar de EPC, este es el resultado final.

2. De lo contrario, aplique el procedimiento de la Sección 12.3.3 al URI de etiqueta de EPC del paso 1 para obtener un URI de EPC de identidad pura. Este es el resultado final.

15.2.4 Decodificación de información de control

Este procedimiento se usa como una subrutina de los procedimientos de decodificación en las secciones 15.2.1 y 15.2.2. Con él, se calcula una cadena que se inserta de inmediato después del prefijo `urn:epc:tag:` o `urn:epc:raw:`, que contiene los valores de todos los campos de información de control que no contienen ceros (además del valor de filtro). Si todos los campos contienen ceros, este procedimiento tiene como resultado una cadena vacía, en cuyo caso nada adicional se inserta después del prefijo `urn:epc:tag:` o `urn:epc:raw:`.

Datos:

- Contenido del banco de memoria de EPC de una etiqueta Gen 2

Procedimiento:

1. Si el bit 17h es cero, extraiga los bits 18h - 1Fh y considérelos como un entero sin signos *A*. Si *A* no es un cero, prefije la cadena `[att=xAA]` (incluidos los corchetes) a *CF*, donde *AA* es el valor de *A* como un numeral hexadecimal de dos dígitos.

2. Si el bit 15h no es cero, fija la cadena `[umi=i]` (incluidos los corchetes) a *CF*.

3. Si el bit 16h no es cero, extraiga los bits 210h - 21Fh y considérelos como un entero sin signos *X*. Si *X* no es un cero, prefije la cadena `[xpc=xXXXX]` (incluidos los corchetes) a *CF*, donde *XXXX* es el valor de *X* como un numeral hexadecimal de cuatro dígitos. Tome en cuenta que la interfaz aérea Gen 2, los bits 210h - 21Fh se insertan en los datos de inventario de retrodispersión inmediatamente después del bit 1Fh, cuando el bit 16h no es cero. Consulte [UHFC1G2].

4. Genere la cadena resultante (que debe estar vacía).

16 Contenido del banco de memoria de identificación de etiqueta (TID)

Para cumplir con esta especificación, el banco de memoria de identificación de etiqueta (banco 10) DEBE contener un identificador de clase de asignación ISO/IEC 15963 de 8 bits de E2h en las localizaciones de la memoria 00h a 07h. La memoria de TID arriba de la localización 07h SE DEBE configurar como se indica a continuación:

- 08_h: Indicador XTID (**X**) (donde una etiqueta implementa una identificación de etiqueta extendida, XTID)
- 09_h: Indicador de seguridad (**S**) (donde una etiqueta admite los comandos *Authenticate* y/o *Challenge*)
- $0A_h$: Indicador de archivo (**F**) (si una etiqueta admite el comando ***FileOpen***)
- $0B_h$ a 13_h: identificador de diseño de enmascaramiento de 9 bits (**MDID**) disponible a partir de GS1
- 14h a $1F_h$: un número de modelo de etiqueta definido por el fabricante de la etiqueta, de 12 bits (**TMN**)
- Arriba de $1F_h$: según se define en la sección 16.2 más abajo

Al número de modelo de etiqueta (TMN) se le puede asignar cualquier valor del titular de un MDID determinado. Sin embargo, en [UHFC1G2] se indica que "se deben definir localizaciones en la memoria de TID arriba de 07h de acuerdo con la autoridad de registro definidos por este valor de identificador de clase y deben contener, como mínimo, suficiente información de identificación para un interrogador para identificar los comandos especiales y/o las características opcionales que admite una etiqueta". Para el identificador de clase de asignación de E2h esta información es el MDID y TMN, independientemente de que el TID extendido esté presente o no. Si dos etiquetas difieren en los comandos especiales y/o en las características opcionales, se deben asignar distintas combinaciones de MDID/TMN. En particular, si dos etiquetas contienen un TID extendido y los valores en los TID extendidos respectivos difieren en cualquier valor además del valor del número de serie, se les debe asignar una combinación distinta de MDID/TMN. (Por definición, el número de serie debe ser distinto para dos etiquetas que tienen el mismo MDID y TMN, de modo que la identificación de etiqueta seriada especificada en la Sección *16.3* es mundialmente única.) Para las etiquetas que no contienen un TID extendido, en principio se debe poder usar el MDID y TMN para buscar la misma información que se codificaría en el TID extendido, si estuviera presente en la etiqueta, y de nuevo se debe usar una combinación de MDID/TMN diferente si dos etiquetas difieren en las capacidades como las describiría el TID extendido, si estuviera realmente presente.

16.1 Identificación de etiqueta corta (TID)

Si el indicador XTID ("X" bit 08h del banco de TID) se establece en cero, el banco de TID solo contiene el identificador de clase de asignación, indicadores de XTID ("X"), seguridad ("S") y archivo ("F"), el identificador del diseñador de enmascaramiento (MDID) y el número de modelo de etiqueta (TMN), según se especifica anteriormente. Los lectores y las aplicaciones que no se configuran para manejar el TID extendido tratarán todos los TID como identificaciones de etiqueta corta, independientemente de si el indicador XTID es cero o uno.

Nota: Los mapas de memoria presentados en este documento son idénticos a como se presentan en [UHFC1G2]. La dirección de la palabra más baja comienza en la base del mapa y aumenta conforme se sube en el mapa. La dirección binaria se lee de izquierda a derecha comenzando con el bit cero y terminando con el bit quince. En los campos (MDID, TMN, etc.) descritos en el documento se coloca el bit más significativo (número de bit más alto) en la dirección binaria más baja en la memoria y el bit menos significativo (bit cero) en la dirección binaria más alta en la memoria. Tome el identificador de clase de asignación ISO/IEC 15963 de E2h = 111000102 como ejemplo. El bit más significativo de este campo es uno y reside en la dirección 00h del banco de memoria de TID. El valor del bit menos significativo es un cero y reside en la dirección 07h del banco de memoria de TID. Cuando las etiquetas distribuyen datos en respuesta a un comando de lectura, trasmiten cada palabra comenzando con la dirección binaria cero y terminando con la dirección binaria quince.

Tabla 16-1 Formato de TID corto

MEM DE TID	DIRECCIÓN BINARIA EN UNA PALABRA (en código hexadecimal)															
DIRECCIÓN BINARIA DEL BANCO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
10_h-$1F_h$	MDID[3:0]				NÚMERO DE MODELO DE ETIQUETA [11:0]											

MEM DE TID DIRECCIÓN BINARIA DEL BANCO	DIRECCIÓN BINARIA EN UNA PALABRA (en código hexadecimal)															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00h-0Fh	E2h								X	S	F	MDID [8:4]				

16.2 Identificación de etiqueta extendida (XTID)

La XTID está diseñada para proporcionar más información a los usuarios finales sobre las capacidades de las etiquetas que se observan en las aplicaciones de RFID. La XTID extiende el formato y aumenta el soporte para la serialización e información sobre las características principales que implementa la etiqueta.

Si el bit de XTID (bit 08h del banco de TID) se establece en uno, el banco de TID DEBE contener el identificador de clase de asignación, el identificador del diseñador de enmascaramiento (MDID) y el número de modelo de etiqueta (TMN) según se especifica anteriormente, y también DEBE contener información adicional, según se especifica en esta sección.

Si el bit XTID definido antes es uno, las localizaciones de la memoria de TID 20h a 2Fh DEBEN contener un encabezado de XTID de 16 bits, según se especifica en la Sección *16.2.1*. Los valores en el encabezado de XTID especifican qué información adicional está presente en las localizaciones 30h y superiores de la memoria. Las localizaciones 00h a 2Fh de la memoria de TID son los únicos campos de localización fija en el TID extendido; todos los campos que siguen al encabezado de XTID pueden variar en localización en la memoria según los valores en el encabezado de XTID.

La información en el XTID que sigue al encabezado de XTID DEBE consistir en "segmentos" de cero o de palabras múltiples, cada uno dividido entre uno o más "campos", que proporciona determinada información sobre la etiqueta, tal como se estipula más abajo. El encabezado de XTID indica los segmentos de XTID del diseñador de enmascaramiento de la etiqueta que se ha elegido incluir. El orden de los segmentos de XTID en el banco de TID debe seguir el que se menciona en el encabezado de XTID desde el bit más significativo hasta el bit menos significativo. Si un segmento XTID no está presente, los segmentos en los bits menos significativos en el encabezado de XTID deben moverse a las direcciones de memoria de TID más bajas para mantener la estructura de la memoria de XTID contigua. De esta manera, se usa una cantidad mínima de memoria para proporcionar un número de serie y/o describir las características de la etiqueta. En la tabla de abajo se muestra un XTID totalmente llenado.

No normativo: El encabezado de XTID que corresponde a este mapa de memoria sería 0011110000000000 2. Si la etiqueta solo contenía un número de serie de 48 bits, el encabezado de XTID sería 0010000000000000 2. El número de serie comenzaría en la dirección binaria 30h y terminaría en la dirección binaria 5Fh. Si la etiqueta contenía únicamente el segmento BlockWrite y BlockErase y el segmento User Memory y BlockPermaLock, el encabezado de XTID sería 0000110000000000 2. El segmento BlockWrite y BlockErase comenzaría en la dirección binaria 30h y terminaría en la dirección binaria 6Fh. El segmento User Memory y BlockPermaLock comenzaría en la dirección binaria 70h y terminaría en la dirección binaria 8Fh.

Tabla 16-2 Formato de identificación de etiqueta extendida (XTID) para el banco de memoria de TID. Tome en cuenta que la tabla anterior está totalmente llenada y la cantidad real de memoria usada, la presencia de un segmento y la localización en la memoria de un segmento dependen del encabezado de XTID.

Referencia de TDS Sección	MEM DE TID DIRECCIÓN BINARIA DEL BANCO	DIRECCIÓN BINARIA EN UNA PALABRA (en código hexadecimal)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
16.2.5	C0h-CFh	Segmento User Memory y BlockPermaLock [15:0]															
	B0h-BFh	Segmento User Memory y BlockPermaLock [31:16]															
16.2.4	A0h-AFh	Segmento BlockWrite y BlockErase [15:0]															
	90h-9Fh	Segmento BlockWrite y BlockErase [31:16]															
	80h-8Fh	Segmento BlockWrite y BlockErase [47:32]															
	70h-7Fh	Segmento BlockWrite y BlockErase [63:48]															
16.2.3	60h-6Fh	Segmento de soporte de comando opcional [15:0]															
16.2.2	50h-5Fh	Segmento de número de serie [15:0]															
	40h-4Fh	Segmento de número de serie [31:16]															

Referencia de TDS Sección	BIT DE BANCO DE MEM DE TID DIRECCIÓN	DIRECCIÓN BINARIA EN UNA PALABRA (en código hexadecimal)															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	30_h-$3F_h$	Segmento de número de serie [47:32]															
16.2.1	20_h-$2F_h$	Segmento de encabezado de XTID [15:0]															
16.1	10_h-$1F_h$	Consulte la **Tabla 16-1** Formato de TID corto															
	00_h-$0F_h$																

16.2.1 Encabezado de XTID

El encabezado de XTID se muestra en la *Tabla 16-3*. Contiene bits definidos y reservados para uso futuro (RFU). El bit del encabezado extendido y los bits de RFU (bits 9 a 0) se deben establecer en cero a fin de cumplir con esta versión de la especificación. Los bits 15 a 13 del encabezado de XTID indican la presencia y el tamaño de la serialización de la etiqueta. Si se establecen en cero, no ocurre ninguna serialización en el XTID. Si no son ceros, un número de serie de etiqueta sigue inmediatamente al encabezado. Las características opcionales en los bits 12 a 10 se manejan de manera diferente. Un cero indica que el lector tiene que realizar una búsqueda en la base de datos o que la etiqueta no admite la característica opcional. Un uno indica que la etiqueta admite la característica opcional y que el XTID contiene el segmento que describe esta característica.

Observe que el contenido del encabezado de XTID determina únicamente la longitud total del XTID, así como la dirección de inicio de cada segmento de XTID incluido.

Tabla 16-3 Encabezado de XTID

Posición binaria en la palabra	Campo	Descripción
0	Extended Header Present	Si no contiene cero, especifica que los bits del encabezado de XTID adicionales están presentes después de los 16 bits del encabezado de XTID que se especifican aquí. Esto proporciona un mecanismo para extender el XTID en versiones futuras del Estándar de Datos de Etiquetas de EPC. Este bit se DEBE establecer en cero para cumplir con esta versión del Estándar de Datos de Etiquetas de EPC. Si es cero, especifique que el encabezado de XTID únicamente contiene los 16 bits definidos aquí.
9 - 1	RFU	Reservado para uso futuro. Estos bits DEBEN ser ceros para que cumplan con esta versión del Estándar de Datos de Etiquetas de EPC.
10	User Memory and Block Perma Lock Segment Present	Si no son ceros, especifica que el XTID incluye el segmento User Memory and Block PermaLock especificado en la Sección *16.2.5*. Si son ceros, especifica que el XTID no incluye las palabras User Memory y Block PermaLock.
11	BlockWrite and BlockErase Segment Present	Si no son ceros, especifica que el XTID incluye el segmento BlockWrite and BlockErase especificado en la Sección *16.2.4*. Si son ceros, especifica que el XTID no incluye las palabras BlockWrite y BlockErase.
12	Optional Command Support Segment Present	Si no son ceros, especifica que el XTID incluye el segmento Optional Command Support especificado en la Sección *16.2.3*. Si son ceros, especifica que el XTID no incluye las palabras Optional Command Support.
13 - 15	Serialisation	Si no son ceros, especifica que el XTID incluye un número de serie único, cuya longitud en bits es 48 + 16(*N* - 1), donde *N* es el valor de este campo. Si son ceros, especifica que el XTID no incluye un número de serie único.

16.2.2 Serialización de XTID

La longitud de la serialización de XTID se especifica en el encabezado del XTID. La entidad administrativa que especifica el ID del diseñador de enmascaramiento de etiquetas es responsable de asignar números de serie únicos para cada número de modelo de etiqueta. La longitud del número de serie utiliza el algoritmo siguiente:

0: Indica la ausencia de serialización

1-7: Longitud en bits = 48 + ((Valor-1) * 16)

16.2.3 Segmento de soporte para comando opcional

Si se establece el bit doce en el encabezado del XTID, se añade la palabra siguiente al XTID. Los campos de bit que se dejan como ceros indican que la etiqueta no admite esa funcionalidad. La descripción de las funcionalidades es la siguiente.

Tabla 16-4 Texto del XTID Optional Command Support

Posición binaria en el segmento	Campo	Descripción
4 - 0	Max EPC Size	Este campo de bit cinco debe indicar el tamaño máximo que se puede programar en los primeros cinco bits en la PC.
5	Recom Support	Si se establece este bit, la etiqueta admite la repetición de la puesta en servicio como se especifica en [UHFC1G2].
6	Access	Si se establece este bit, indica que la etiqueta admite el comando access.
7	Separate Lockbits	Si se establece este bit, significa que la etiqueta admite bits de bloqueo para cada banco de memoria en lugar de la implementación más sencilla de un bit de bloqueo único para toda la etiqueta.
8	Auto UMI Support	Si se establece este bit, significa que la etiqueta establece automáticamente su bit indicador de memoria de usuario en la palabra PC.
9	PJM Support	Si se establece este bit, se indica que la etiqueta admite la modulación de fluctuación de fase. Este es un modo de modulación opcional que únicamente soportan las etiquetas de HF Gen 2.
10	BlockErase Supported	Si se establece, indica que la etiqueta admite el comando BlockErase. La manera en que la etiqueta admite el comando BlockErase se describe en la Sección *16.2.4*. Un fabricante puede elegir establecer este bit, pero no incluir el campo BlockWrite and BlockErase si se debe explicar detalladamente cómo usar el comando a través de una búsqueda en la base de datos.
11	BlockWrite Supported	Si se establece, indica que la etiqueta admite el comando BlockWrite. La manera en que la etiqueta admite el comando BlockErase se describe en la Sección *16.2.4*. Un fabricante puede elegir establecer este bit, pero no incluir el campo BlockWrite and BlockErase si se debe explicar detalladamente cómo usar el comando a través de una búsqueda en la base de datos.
12	BlockPermaLock Supported	Si se establece, indica que la etiqueta admite el comando BlockPermaLock. La manera en que la etiqueta admite el comando BlockPermaLock se describe en la Sección *16.2.5*. Un fabricante puede elegir establecer este bit, pero no incluir el campo BlockPermaLock and User Memory si se debe explicar detalladamente cómo usar el comando a través de una búsqueda en la base de datos.
15 - 13	[RFU]	Estos bits están RFU y se deben establecer en cero.

16.2.4 Segmento BlockWrite y BlockErase

Si se establece el bit once del encabezado de XTID, el XTID debe incluir el segmento BlockWrite and BlockErase de cuatro palabras. Para indicar que no admite un comando determinado, la etiqueta debe tener todos los campos relacionados con ese comando establecidos en cero. Este siempre DEBE ser el caso cuando esté presente el segmento Optional Command Support (Sección *16.2.3*) e indica que no se admite el comando BlockWrite o BlockErase. A continuación se incluye una descripción de los campos.

Tabla 16-5 Información del XTID Block Write and Block Erase

Posición binaria en el segmento	Campo	Descripción
7 - 0	Block Write Size	Tamaño máximo del bloque que admite la etiqueta para el comando BlockWrite. Este valor debe ser entre 1-255 si se describe el comando BlockWrite en este campo.
8	Variable Size Block Write	Este bit se usa para indicar si la etiqueta admite los comandos BlockWrite con bloques de tamaño variable. Si el valor es cero, la etiqueta solo admite bloques de escritura exactamente del tamaño máximo del bloque que se indica en bits [7-0]. Si el valor es uno, la etiqueta admite bloques de escritura cuyo tamaño es menor al máximo indicado en bits [7-0].
16 - 9	Block Write EPC Address Offset	Esto indica la dirección de la palabra inicial del primer bloque completo en el que se puede escribir usando BlockWrite en el banco de memoria de EPC.

Posición binaria en el segmento	Campo	Descripción
17	No Block Write EPC address alignment	Este bit se usa para indicar si la arquitectura de la memoria de etiquetas tiene límites rígidos para los bloques en el banco de memoria de EPC. Si el valor es cero, la etiqueta tiene límites rígidos para bloques en el banco de memoria de EPC. La etiqueta no aceptará comandos BlockWrite que comiencen en un bloque y finalicen en otro. Estos límites de bloque se determinan mediante el tamaño máximo del bloque y la dirección de inicio del primer bloque completo. Todos los bloques tienen el mismo tamaño máximo. Si el valor es uno, la etiqueta no tiene límites rígidos para bloques en el banco de memoria de EPC. Aceptará todos los comandos BlockWrite que estén en el banco de memoria.
25 - 18	Block Write User Address Offset	Esto indica la dirección de la palabra inicial del primer bloque completo en el que se puede escribir usando BlockWrite en la memoria del usuario.
26	No Block Write User Address Alignment	Este bit se usa para indicar si la arquitectura de la memoria de etiquetas tiene límites rígidos para los bloques en el banco de memoria del USUARIO. Si el valor es cero, la etiqueta tiene límites rígidos para bloques en el banco de memoria del USUARIO. La etiqueta no aceptará comandos BlockWrite que comiencen en un bloque y finalicen en otro. Estos límites de bloque se determinan mediante el tamaño máximo del bloque y la dirección de inicio del primer bloque completo. Todos los bloques tienen el mismo tamaño máximo. Si el valor es uno, la etiqueta no tiene límites rígidos para bloques en el banco de memoria del USUARIO. Aceptará todos los comandos BlockWrite que estén en el banco de memoria.
31 - 27	[RFU]	Estos bits están RFU y se deben establecer en cero.
39 -32	Size of Block Erase	Tamaño máximo del bloque que admite la etiqueta para el comando BlockErase. Este valor debería estar entre 1-255 si se describe el comando BlockErase en este campo.
40	Variable Size Block Erase	Este bit se usa para indicar si la etiqueta admite los comandos BlockErase con bloques de tamaño variable. Si el valor es cero, la etiqueta solo admite bloques de borrado exactamente del tamaño máximo del bloque que se indica en bits [39-32]. Si el valor es uno, la etiqueta admite bloques de borrado cuyo tamaño es menor al máximo indicado en bits [39-32].
48 - 41	Block Erase EPC Address Offset	Esto indica la dirección inicial del primer bloque completo en el que se puede borrar en el banco de memoria de EPC.
49	No Block Erase EPC Address Alignment	Este bit se usa para indicar si la arquitectura de la memoria de etiquetas tiene límites rígidos para los bloques en el banco de memoria de EPC. Si el valor es cero, la etiqueta tiene límites rígidos para bloques en el banco de memoria de EPC. La etiqueta no aceptará comandos BlockErase que comiencen en un bloque y finalicen en otro. Estos límites de bloque se determinan mediante el tamaño máximo del bloque y la dirección de inicio del primer bloque completo. Todos los bloques tienen el mismo tamaño máximo. Si el valor es uno, la etiqueta no tiene límites rígidos para bloques en el banco de memoria de EPC. Aceptará todos los comandos BlockErase que estén en el banco de memoria.
57 - 50	Block Erase User Address Offset	Esto indica la dirección inicial del primer bloque completo en el que se puede borrar en el banco de memoria del usuario.
58	No Block Erase User Address Alignment	Bit 58: Este bit se usa para indicar si la arquitectura de la memoria de etiquetas tiene límites rígidos para los bloques en el banco de memoria del USUARIO. Si el valor es cero, la etiqueta tiene límites rígidos para bloques en el banco de memoria del USUARIO. La etiqueta no aceptará comandos BlockErase que comiencen en un bloque y finalicen en otro. Estos límites de bloque se determinan mediante el tamaño máximo del bloque y la dirección de inicio del primer bloque completo. Todos los bloques tienen el mismo tamaño máximo. Si el valor es uno, la etiqueta no tiene límites rígidos para bloques en el banco de memoria del USUARIO. Aceptará todos los comandos BlockErase que estén en el banco de memoria.
63 - 59	[RFU]	Estos bits están reservados para uso futuro y deberían establecer en cero.

16.2.5 Segmento Memoria de usuario y BlockPermaLock

Este segmento de dos palabras está presente en el XTID si se establece el bit 10 del encabezado de XTID. Los bits 15 - 0 deben indicar el tamaño de la memoria del usuario en palabras. Los bits 31-16 deben indicar el tamaño de los bloques en el banco de memoria del USUARIO en palabras para el comando BlockPermaLock. Nota: Estos tamaños de bloque solo se aplican al comando BlockPermaLock y son independientes de los comandos BlockWrite and BlockErase.

Tabla 16-6 Información del XTID Block PermaLock and User Memory

Posición binaria en el segmento	Campo	Descripción
15 - 0	User Memory Size	Cantidad de palabras de 16 bits en la memoria del usuario.
31 -16	BlockPermaLock Block Size	Si no son ceros, el tamaño en palabras de cada bloque que se puede bloquear de manera permanente. Es decir, la funcionalidad de bloqueo permanente de bloques permite bloquear los bloques de *N**16 bits, donde ***N*** es el valor de este campo. Si son ceros, el XTID no incluye el tamaño del bloque para la funcionalidad BlockPermaLock. La etiqueta puede soportar o no el bloqueo permanente de bloques. Este campo DEBE ser cero si el segmento Optional Command Support (Sección *16.2.3)* está presente y si el bit de BlockPermaLockSupported es cero.

16.3 Identificación serializada de etiqueta (STID)

Esta sección especifica una forma de URI para la serialización que se codifica en un XTID, llamado identificador de etiqueta serializada (STID). Las aplicaciones empresariales que usan un TID serializado pueden usar la forma de URI de STID para identificar de manera única la etiqueta en la que se ha programado un EPC. El URI de STDI está diseñado para complementar, no reemplazar, el EPC de las aplicaciones que utilizan la serialización de etiquetas de RFID además del EPC que identifica de manera única el objeto físico al que se fija la etiqueta; por ejemplo, en una aplicación que utiliza el STID para ayudar a garantizar que no se haya falsificado una etiqueta.

16.3.1 Sintaxis del URI de STID

La sintaxis del URI de STID URI se especifica con la sintaxis siguiente:

```
STID-URI ::= "urn:epc:stid:" 2*( "x" HexComponent "." ) "x" HexComponent
```

Donde el primer y segundo HexComponents DEBEN constar de exactamente tres UpperHexChars y el tercer HexComponent DEBE constar de 12, 16, 20, 24, 28, 32 o 36 UpperHexChars.

El primer HexComponent es el valor del ID del diseñador de enmascaramiento de etiquetas (MDID) según se especifica en la Sección *16.1*. El segundo `HexComponent` es el valor del número de modelo de etiqueta, según se especifica en las secciones *16.1*. El tercer `HexComponent` es el valor del número de serie de XTID, según se especifica en las secciones *16.2.1* y *16.2.2.* La cantidad de UpperHexChars en el tercer `HexComponent` es igual a la cantidad de bits en el número de serie de XTID dividido entre cuatro.

16.3.2 Procedimiento de decodificación: Contenido del banco de TID en el URI de STID

El procedimiento siguiente especifica cómo construir un URI de STID dado el contenido del banco de TID de una etiqueta Gen 2.

Datos:

- Contenido del banco de memoria de TID de una etiqueta Gen 2, como una cadena de bits *b*0*b*1.*b*N-1, donde la cantidad de bits N es por lo menos de 48.

Resultado:

- Un URI de STID

Procedimiento:

1. Los bits $b_0.b_7$ deben coincidir con el valor 11100010. Si no, deténgase: el contenido de este banco de TID no incluye un XTID, según se especifica a continuación.

2. El bit *b*8 se debe establecer en uno. Si no, deténgase: el contenido de este banco de TID no incluye un XTID, según se especifica a continuación.
3. Considere los bits *b*8...*b*19 como un entero sin signo de 12 bits. Este es el ID del diseñador de enmascaramiento de etiqueta (MDID).
4. Considere los bits *b*20...*b*31 como un entero sin signo de 12 bits. Este es el número de modelo de etiqueta.
5. Considere los bits *b*32…*b*34 como un entero V sin signo de 3 bits. Si V equivale a cero, deténgase: el contenido de este banco de TID no incluye un número de serie. De otro modo, calcule la longitud del número de serie L = 48 + 16(V - 1). Considere los bits *b*48*b*49...*b*48+L-1 como un entero sin signo de bits L. Este es el número de serie.
6. Construya el STID-URI concatenando las cadenas siguientes: el prefijo `urn:epc:stid:`, la letra minúscula x, el valor del MDID del paso 3 como un numeral hexadecimal de 3 caracteres, un carácter de punto (.), la letra minúscula x, el valor del número de modelo de etiqueta del paso 4 como un numeral hexadecimal de 3 caracteres, un carácter de punto (.), la letra minúscula x y el valor del número de serie del paso 5 como un numeral hexadecimal de caracteres (L/4). Solo se deben usar las letras mayúsculas A a F al construir los numerales hexadecimales.

17 Contenido del banco de memoria del usuario

El banco de memoria del usuario de EPCglobal proporciona una memoria de tamaño variable para almacenar atributos de datos adicionales relacionados con el objeto identificado en el banco de memoria de EPC de la etiqueta.

La memoria del usuario puede estar presente o no en una etiqueta determinada. Cuando la memoria del usuario no está presente, el bit 15h del banco de memoria de EPC se DEBE establecer en cero. Cuando la memoria del usuario está presente y no se inicializa, el bit 15h del banco de memoria de EPC se DEBE establecer en cero y los bits 03h a 07h del banco de memoria del usuario se DEBEN establecer en cero. Cuando la memoria del usuario está presente y se inicializa, el bit 15h de la palabra de control del protocolo en la memoria de EPC se DEBE establecer en uno para indicar la presencia de datos codificados en la memoria del usuario, y el banco de memoria del usuario se DEBE programar, según se especifica aquí.

A fin de cumplir con esta especificación, los primeros ocho bits del banco de memoria de usuario DEBE contener un identificador de formato de almacenamiento de datos (DSFID), según se especifica en [ISO15962]. Esto mantiene la compatibilidad con otros estándares. El DSFID consta de tres campos lógicos: Access Method, Extended Syntax Indicator y Data Format. Access Method se especifica en los dos bits más significativos del DSFID, y se codifica con el valor "10" para designar el método de acceso "Packed Objects" (objetos empaquetados), según se indica en el Apéndice I si se emplea el método de acceso "Packed Objects", y se codifica con el valor "00" para designar el método de acceso "No-Directory" (sin directorio) tal como se indica en [ISO15962] si se emplea el método de acceso "No-Directory". El siguiente bit se establece en uno si hay un segundo byte de DSFID presente. Los cinco bits menos significativos especifican el formato de datos, que indica cuál sistema de datos predomina en el contenido de la memoria. Si los identificadores de aplicación (AI) GS1 predominan, el valor "01001" especifica el formato de datos GS1 09 como registrado con ISO, lo que proporciona el soporte más eficiente para el uso de elementos de datos de AI. Los apéndice I a M de esta especificación contienen la especificación completa del método de acceso "Packed Objects"; se espera que este contenido aparezca como Anexo *I* a *M*, respectivamente, de ISO/IEC 15962, 2ª edición [ISO15962], cuando esta última sea publicada. Se incluye una definición completa del DSFID en ISO/IEC 15962 [ISO15962]. Se incluye una definición completa de la tabla que regula los objetos empaquetados que codifican los identificadores de aplicación (AI), tal como se especifica en GS1 y se registró con ISO conforme a los procedimientos de ISO/IEC 15961, y se reproduce en *E.3*. Esta tabla es similar en formato al ejemplo hipotético que se muestra en la Tabla L-1 en *L*, pero con entradas que permiten la codificación de todos los identificadores de aplicación válidos.

Se DEBE codificar una etiqueta cuya programación en el banco de memoria del usuario cumpla con esta especificación, ya sea mediante el método de acceso de objetos empaquetados o el método de acceso sin directorio, siempre y cuando se utilice el método de acceso sin directorio, NO DEBE usarse el modo de concentración "definido por la aplicación", según se especifica en [ISO15962]. Una etiqueta cuya programación en el banco de memoria del usuario cumple con esta especificación PUEDE usar cualquier formato de datos registrado, incluido el formato de datos 09.

Cuando la especificación de objetos empaquetados en *I* haga referencia a vectores de bits ampliables (EBV), se DEBE usar el formato especificado en el Apéndice *D*.

Un componente de hardware o software que cumpla con esta especificación para la lectura y escritura en el banco de memoria del usuario DEBE implementar por completo el método de acceso de objetos empaquetados, según se especifica en los Apéndices *I* a *M* de esta especificación (lo que implica el soporte de todos los formatos de datos registrados), DEBE implementar el método de acceso sin directorio tal como se indica en [ISO15962] y PUEDE implementar otros métodos de acceso definidos en [ISO15962] y en las versiones posteriores de ese estándar.

Sin embargo, un componente de hardware o software NO NECESITA implementar el modo de concentración "definido por la aplicación" del método de acceso sin directorio especificado en [ISO15962]. Un componente de hardware o software cuya función ideal es únicamente inicializar etiquetas (por ejemplo, una impresora) puede cumplir con un subconjunto de esta especificación al implementar el método de acceso de objetos empaquetados o el método de acceso sin directorio, pero en este caso NO NECESITAN implementarse ambos.

Reglas no normativas: Explicación: Esta especificación permite dos métodos de codificación de datos en la memoria del usuario. El método de acceso "No-Directory" de ISO/IEC 15962 tiene una base instalada debido a sus amplios antecedentes y niveles de aceptación en determinadas comunidades de usuarios finales. El método de acceso de objetos empaquetados se desarrolló para mejorar la lectura y escritura de etiquetas, y disminuir el consumo de la memoria de etiquetas.

El modo de concentración "definido por la aplicación" del método de acceso sin directorio no está permitido debido a que un sistema de recepción no lo puede comprender a menos de que ambos lados tengan la misma definición de cómo funciona la concentración.

Tome en cuenta que el método de acceso de objetos empaquetados admite la codificación de datos con o sin una estructura similar a un directorio para el acceso aleatorio. El hecho de que el otro método de acceso se llame "No-Directory" en [ISO15962] no debería interpretarse como que implique que el método de acceso de objetos empaquetados siempre incluye un directorio.

18 Cumplimiento

Por naturaleza, el Estándar de Datos de Etiquetas EPC tiene impacto en varias partes del Marco de la Arquitectura de EPCglobal. A diferencia de otros estándares que definen una interfaz específica de hardware o software, el Estándar de Datos de Etiquetas EPC define formatos de datos, junto con procedimientos para convertir entre formatos equivalentes. Tanto los formatos de datos como los procedimientos de conversión se emplean en varios componentes de hardware, software y datos en cualquier sistema determinado.

En esta sección se define lo que significa cumplir con el Estándar de Datos de Etiquetas EPC. Como se indica más arriba, hay varios tipos de componentes del sistema que pueden cumplir con varias partes del Estándar de Datos de Etiquetas EPC, y se enumeran más abajo.

18.1 Cumplimiento de los datos de etiqueta RFID

Los datos programados en una etiqueta RFID Gen 2 puede cumplir con el Estándar de Datos de Etiquetas EPC según se especifica más abajo. El cumplimiento se puede evaluar por separado para el contenido de cada banco de memoria.

Cada banco de memoria puede estar en estado "sin inicializar" o en estado "inicializado". El estado sin inicializar indica que el banco de memoria no contiene datos y, por lo general, solo se usa entre el momento en que se elabora una etiqueta y el momento en que una aplicación lo programa por primera vez para su uso. Los requisitos de cumplimiento se proporcionan por separado para cada estado, según corresponda.

18.1.1 Cumplimiento del banco de memoria reservado (banco 00)

El contenido del banco de memoria reservado (banco 00) de una etiqueta Gen 2 no está sujeto al cumplimiento del Estándar de Datos de Etiquetas EPC. El contenido del banco de memoria reservado se especifica en [UHFC1G2].

18.1.2 Cumplimiento del banco de memoria de EPC (banco 01)

El contenido del banco de memoria de EPC (banco 01) de una etiqueta Gen 2 no está sujeto al cumplimiento del Estándar de Datos de Etiquetas EPC, según se indica a continuación.

El contenido del banco de memoria de EPC cumple con el Estándar de Datos de Etiquetas EPC en el estado sin inicializar si todas las afirmaciones siguientes son verdaderas:

- El bit 17_h se DEBE establecer en cero.
- Los bits 18_h a 1Fh (inclusive), los bits de atributo, se DEBEN establecer en cero.
- Los bits 20_h a 27_h (inclusive) se DEBEN establecer en cero, lo que indica un banco de memoria de EPC sin inicializar.
- Todos los demás bits del banco de memoria de EPC DEBEN aparecer como se especifica en la Sección 9 y/o en [UHFC1G2], según corresponda.

El contenido del banco de memoria de EPC cumple con el Estándar de Datos de Etiquetas EPC en el estado inicializado si todas las afirmaciones siguientes son verdaderas:

- El bit 17_h se DEBE establecer en cero.

- Los bits 18h a 1Fh (inclusive), los bits de atributo, DEBEN aparecer como se especifica en la Sección 11.
- Los bits 20h a 27h (inclusive) se DEBEN establecer en un valor de encabezado de EPC válido, según se especifica en la Tabla 14-1; es decir, un valor de encabezado no marcado como "reservado" o "etiqueta no programada" en la tabla.
- Deje N como el valor de la columna "longitud de codificación" de la fila de la Tabla 14-1 que corresponde al valor del encabezado, y M debe ser igual a 20h + N - 1. Los bits 20h a M DEBEN ser un valor de codificación binaria de EPC válido; es decir, el procedimiento de decodificación de la Sección 14.3.7 cuando se aplica a estos bits NO DEBE generar una excepción.
- Los bits M+1 hasta el final del banco de memoria de EPC o el bit 20Fh (el que aparezca primero) se DEBEN establecer en cero.
- Todos los demás bits del banco de memoria de EPC DEBEN aparecer como se especifica en la Sección 9 y/o en [UHFC1G2], según corresponda.

No normativo: Explicación: Una consecuencia de los requisitos anteriores es que para cumplir con esta especificación, no se pueden incluir datos de aplicación adicionales (como un segundo EPC) en el banco de memoria de EPC después del EPC que comienza en el bit 20h.

18.1.3 Cumplimiento del banco de memoria de EPC (banco 10)

El contenido del banco de memoria de TID (banco 10) de una etiqueta Gen 2 no está sujeto al cumplimiento del Estándar de Datos de Etiquetas EPC, según se especifica en la Sección 16.

18.1.4 Cumplimiento del banco de memoria del usuario (banco 11)

El contenido del banco de memoria del usuario (banco 11) de una etiqueta Gen 2 no está sujeto al cumplimiento del Estándar de Datos de Etiquetas EPC, según se especifica en la Sección 17.

18.2 Cumplimiento de los componentes de hardware y software

Los componentes de hardware y software pueden procesar datos que se leen o escriben en las etiquetas RFID Gen 2. Los componentes de hardware y software también pueden manipular códigos de productos electrónicos en varias formas independientemente de si hay etiquetas RFID. Todos estos usos pueden estar sujetos al cumplimiento con el Estándar de Datos de Etiquetas EPC según se especifica más adelante. Lo que se requiere exactamente para cumplir con los requisitos depende de la función deseada o supuesta del componente de hardware o software.

18.2.1 Cumplimiento de los componentes de hardware y software que producen o consumen contenido del banco de memoria Gen 2

En esta sección se especifica el cumplimiento de los componentes de hardware y software que producen y consumen el contenido de un banco de memoria de una etiqueta Gen 2. Esto incluye los componentes que interactúan directamente con etiquetas a través de la interfaz aérea Gen 2, así como los componentes que manipulan una representación de software de contenido de memoria sin procesar.

Definiciones:

- **Consumidor de banco X** (donde X es un banco de memoria específico de una etiqueta Gen 2): Un componente de hardware o software que acepta como entrada el contenido de un banco X de una etiqueta Gen 2 a través de una interfaz externa. Esto incluye componentes que leen etiquetas a través de la interfaz aérea Gen 2 (es decir, lectores), así como componentes que manipulan una representación de software de contenido de memoria sin procesar (por ejemplo, software de "middleware" que recibe una imagen en formato hexadecimal de la memoria de etiquetas de un interrogador como entrada).

- **Productor de banco X** (donde X es un banco de memoria específico de una etiqueta Gen 2): Un componente de hardware o software que genera el contenido de un banco X Gen 2 a través de una interfaz externa. Esto incluye componentes que interactúan directamente con etiquetas a través de la interfaz aérea Gen 2 (es decir, interrogadores de escritura e impresoras, el contenido de la memoria que se incluye en la etiqueta es una salida de la interfaz aérea), así como los componentes que manipulan una representación de software de contenido de memoria sin procesar (por ejemplo, software que produce un comando de "escritura" para un interrogador, que genera una imagen en formato hexadecimal de la memoria de etiquetas como parte del comando).

Un componente de hardware o software que "pasa" el contenido sin procesar del banco X de memoria de etiquetas de una interfaz externa a otra es al mismo tiempo un consumidor del banco X y un productor del banco X. Por ejemplo, considere un dispositivo de lectura que acepta como entrada de una aplicación a través del "protocolo electrónico" de red un comando para escribir una memoria de etiquetas de EPC, donde el comando incluye una imagen con formato hexadecimal de la memoria de etiquetas en la aplicación en la que quiere escribir y luego escribe la imagen en una etiqueta a través de la interfaz aérea Gen 2. Ese dispositivo es un consumidor del banco 01 respecto del "protocolo electrónico" y un productor del banco 01 respecto de la interfaz aérea Gen 2. Los requisitos de cumplimiento siguientes garantizan que ese dispositivo pueda aceptar de una aplicación y escribir en una etiqueta cualquier contenido del banco de EPC que sea válido, de acuerdo con esta especificación.

Los requisitos de cumplimiento siguientes se aplican a los consumidores y productores del banco X, según se define a continuación:

- Un consumidor del banco 01 (banco de EPC) DEBE aceptar como entrada cualquier contenido de la memoria que cumpla con esta especificación, según se especifica en la Sección *18.1.2*.
- Si un consumidor del banco 01 interpreta el contenido del banco de memoria de EPC recibido como entrada, DEBE hacerlo de una manera compatible con las definiciones del contenido del banco de memoria de EPC de esta especificación.
- Un productor del banco 01 (banco de EPC) DEBE producir contenido de memoria que cumpla con esta especificación, según se especifica en la Sección *18.1.2*, siempre que el componente de hardware o software genere datos de salida para el banco 01 que contiene un EPC. Un productor del banco 01 PUEDE producir datos de salida que no contengan un EPC si el bit 17h se establece en el valor uno.
- Si un productor del banco 01 construye el contenido del banco de memoria de EPC a partir de las partes del componente, DEBE hacerlo de una manera compatible con lo indicado.
- Un consumidor del banco 10 (banco de TID) DEBE aceptar como entrada cualquier contenido de la memoria que cumpla con esta especificación, según se especifica en la Sección *18.1.3*.
- Si un consumidor del banco 10 interpreta el contenido del banco de memoria de TID recibido como entrada, DEBE hacerlo de una manera compatible con las definiciones del contenido del banco de memoria de TID de esta especificación.
- Un productor del banco 10 (banco de TID) DEBE producir contenido de la memoria que cumpla con esta especificación, según se especifica en la Sección *18.1.3*.
- Si un productor del banco 10 construye el contenido del banco de memoria de TID a partir de las partes del componente, DEBE hacerlo de una manera compatible con esta especificación.
- El cumplimiento para los componentes de hardware o software que lean o escriban el banco de memoria del usuario (banco 11) DEBE realizarse según se especifica en la Sección *17*.

18.2.2 Cumplimiento de los componentes de hardware y software que producen o consumen formas de URI del EPC

En esta sección se especifica el cumplimiento de los componentes de hardware y software que usa URI, según se especifica aquí como entradas o salidas.

Definiciones:

- **Consumidor de URI de EPC**: Componente de hardware o software que acepta un URI de EPC como entrada a través de una interfaz externa. Un consumidor de URI de EPC se puede clasificar como un consumidor de URI de EPC de identidad pura si acepta un URI de identidad pura de EPC como entrada, o un consumidor de URI de etiquetas/datos sin procesar de EPC si acepta un URI de etiqueta de EPC o URI de datos sin procesar de EPC como entrada.

- **Productor de URI de EPC**: Componente de hardware o software que acepta un URI de EPC como salida a través de una interfaz externa. Un consumidor de URI de EPC se puede clasificar como un consumidor de URI de EPC de identidad pura si acepta un URI de identidad pura de EPC como entrada, o un consumidor de URI de etiquetas/datos sin procesar de EPC si acepta un URI de etiqueta de EPC o URI de datos sin procesar de EPC como entrada.

Un componente de hardware o software determinado puede cumplir con más de una de las definiciones anteriores, en cuyo caso está sujeto a todas las pruebas de cumplimiento pertinentes que se describen más abajo.

Los requisitos de cumplimiento siguientes se aplican a los consumidores de EPC de URI de identidad pura:

- Un EPC de URI de identidad pura DEBE aceptar como entrada cualquier cadena que satisfaga la sintaxis de la Sección *6*, incluidas todas las restricciones en la cantidad de caracteres en varios componentes.
- Un EPC de URI de identidad pura DEBE rechazar como inválida cualquier cadena de entrada que comience con los caracteres `urn:epc:id:` que no satisfagan la sintaxis de la Sección *6*, incluidas todas las restricciones en la cantidad de caracteres en varios componentes.
- Si un consumidor de EPC de URI de identidad pura interpreta el contenido de un URI de identidad pura, DEBE hacerlo de una manera compatible con las definiciones del URI de EPC de identidad pura en esta especificación y en las especificaciones que se mencionan aquí (incluidas las Especificaciones Generales de GS1).

Los requisitos de cumplimiento siguientes se aplican a los productores de EPC de URI de identidad pura:

- Un productor de URI de EPC de identidad pura DEBE producir como salida cadenas que satisfagan la sintaxis de la Sección 6, incluidas todas las restricciones en la cantidad de caracteres en varios componentes.
- Un productor de URI de EPC de identidad pura NO DEBE producir como salida una cadena que comience con los caracteres `urn:epc:id:` que no satisfagan la sintaxis de la Sección *6*, incluidas todas las restricciones en la cantidad de caracteres en varios componentes.
- Si un productor de URI de EPC de identidad pura construye un URI de EPC de identidad pura a partir de las partes del componente, DEBE hacerlo de una manera compatible con esta especificación.

Los requisitos de cumplimiento siguientes se aplican a los consumidores de URI de etiquetas/datos sin procesar de EPC:

- Un consumidor de URI de etiquetas/datos sin procesar de EPC DEBE aceptar como entrada la producción de `TagURI` de la sintaxis de la Sección *12.4*, y que se puedan codificar de acuerdo con la Sección 14.3 sin causar una excepción.
- Un consumidor de URI de etiquetas/datos sin procesar de EPC PUEDE aceptar como entrada cualquier cadena que satisfaga la producción de `RawURI` de la sintaxis de la Sección *12.4*.
- Un consumidor de URI de etiquetas/datos sin procesar de EPC DEBE rechazar como inválida cualquier cadena de entrada que comience con los caracteres `urn:epc:tag:` que no satisfagan la sintaxis de la Sección *12.4*, o que provoque que el procedimiento de codificación de la Sección 14.3 genere una excepción.
- Un consumidor de URI de etiquetas/datos sin procesar de EPC que acepte URI de datos sin procesar de EPC como entrada DEBE rechazar como inválida cualquier cadena de entrada que comience con los caracteres `urn:epc:raw:` que no satisfagan la sintaxis de la Sección *12.4*.
- En la medida en que un consumidor de URI de etiquetas/datos sin procesar de EPC interprete el contenido de un URI de etiquetas de EPC o de un URI de datos sin procesar de EPC, DEBE hacerlo de una manera compatible con las definiciones del URI de etiquetas de EPC y el URI de datos sin procesar de EPC en esta especificación y las especificaciones que se mencionan aquí (incluidas las Especificaciones Generales de GS1).

Los requisitos de cumplimiento siguientes se aplican a los productores de URI de etiquetas/datos sin procesar de EPC:

- Un productor de URI de etiquetas/datos sin procesar de EPC DEBE producir como salida cadenas que satisfagan la producción de `TagURI` o la producción de `RawURI` de la sintaxis de la Sección 12.4, siempre que cualquier cadena de salida que satisfaga la producción de `TagURI` deba ser codificada de acuerdo con el procedimiento de codificación de la Sección 14.3 sin causar una excepción.
- Un productor de URI de etiquetas/datos sin procesar de EPC NO DEBE producir como salida una cadena que comience con los caracteres `urn:epc:tag:` o `urn:epc:raw:` según se especificó en la viñeta previa.

- Si un productor de URI de etiquetas/datos sin procesar de EPC construye un URI de etiquetas de EPC o un URI de datos sin procesar de EPC a partir de las partes del componente, DEBE hacerlo de una manera compatible con esta especificación.

18.2.3 Cumplimiento de los componentes de hardware y software que traducen entre formas de EPC

En esta sección se especifica el cumplimiento para los componentes de hardware y software que traducen entre formas de EPC, como traducir una codificación binaria de EPC a un URI de etiqueta de EPC, un URI de etiqueta de EPC a un URI de EPC de identidad pura, un URI de EPC de identidad pura a un URI de etiqueta de EPC, o un URI de etiqueta de EPC al contenido del banco de memoria de EPC de una etiqueta Gen 2. Por definición, cualquier componente acepta estas formas como entradas o salidas y, por lo tanto, está sujeto a las partes relevantes de las secciones *18.2.1* y *18.2.2*.

- Un componente de hardware o software que toma el contenido del banco de memoria de EPC de una etiqueta Gen 2 como entrada y produce el URI de etiqueta de EPC correspondiente o el URI de datos sin procesar de EPC como salida DEBE producir una salida equivalente a la aplicación del procedimiento de decodificación de la Sección *15.2.2* en la entrada.

- Un componente de hardware o software que toma el contenido del banco de memoria de EPC de una etiqueta Gen 2 como entrada y produce el URI de etiqueta de EPC correspondiente o el URI de datos sin procesar de EPC como salida DEBE producir una salida equivalente a la aplicación del procedimiento de decodificación de la Sección *15.2.3* en la entrada.

- Un componente de hardware o software que toma un URI de etiqueta de EPC como entrada y produce el URI de EPC de identidad pura correspondiente como salida DEBE producir una salida equivalente a la aplicación del procedimiento de la Sección *12.3.3* en la entrada.

- Un componente de hardware o software que toma un URI de etiqueta de EPC como entrada y produce el contenido del banco de memoria de EPC de una etiqueta Gen 2 como salida (ya sea escribiendo una etiqueta o produciendo una representación de software del contenido de la memoria de datos sin procesar como salida) DEBE producir una salida equivalente a la aplicación del procedimiento de la Sección *15.1.1* en la entrada.

18.3 Cumplimiento de las formas legibles a simple vista del EPC y del contenido del banco de memoria de EPC

Esta sección especifica el cumplimiento para las representaciones legibles a simple vista de un EPC. Las representaciones legibles a simple vista se pueden usar en etiquetas impresas, en documentos, etc. Esta sección no especifica las condiciones en las que una representación legible a simple vista de un EPC o de una etiqueta RFID debe o debería imprimirse en una etiqueta, un empaque u otro medio; solo especifica cómo es una representación legible a simple vista adecuada cuando se desea incluir una.

- A fin de cumplir con esta especificación, una representación legible a simple vista de un código electrónico de producto DEBE ser un URI de EPC de identidad pura, según se especifica en la Sección *6*.

- Para cumplir con esta especificación, una representación legible a simple vista del contenido completo del banco de memoria de EPC de una etiqueta Gen 2 DEBE ser un URI de etiqueta de EPC o un URI de datos sin procesar de EPC, según se especifica en la Sección *12*. DEBERÍA usar un URI de etiqueta de EPC cuando sea posible hacerlo (es decir, cuando el contenido del banco de memoria contenga un EPC válido).

A Conjunto de caracteres para números de serie alfanuméricos

La tabla siguiente especifica los caracteres permitidos en las Especificaciones Generales de GS1 [GS1GS] para los números de serie alfanuméricos. Las columnas son de la manera siguiente:

- **Símbolo gráfico**: La representación impresa del carácter, tal como se usa en los formatos legibles para los humanos.
- **Nombre**: El nombre común para el carácter
- **Valor hexadecimal**: Un numeral hexadecimal que produce el valor binario de 7 bits para el carácter, tal como se utiliza en las codificaciones binarias de EPC. Este valor hexadecimal siempre es igual al código ISO 646 (ASCII) para el carácter.
- **Forma de URI**: La representación del carácter en las formas de URI de identidad pura EPC y de URI de etiqueta de EPC. Este es un solo carácter cuyos menos significativos del código ASCII son iguales al valor en la columna del "valor hexadecimal" o un triplete de escape que consta de un carácter de porcentaje seguido de dos caracteres que dan el valor hexadecimal al carácter.

Tabla A-1 Caracteres permitidos en números de serie alfanuméricos

Símbolo gráfico	Nombre	Valor hexadecimal	Forma de URI	Símbolo gráfico	Nombre	Valor hexadecimal	Forma de URI
!	Signo de exclamación	21	!	M	Letra mayúscula M	4D	M
"	Comillas	22	%22	N	Letra mayúscula N	4E	N
%	Signo de porcentaje	25	%25	O	Letra mayúscula O	4F	O
&	Ampersand	26	%26	P	Letra mayúscula P	50	P
'	Apóstrofe	27	'	Q	Letra mayúscula Q	51	Q
(	Paréntesis izquierdo	28	(	R	Letra mayúscula R	52	R
)	Paréntesis derecho	29	)	S	Letra mayúscula S	53	S
*	Asterisco	2A	*	T	Letra mayúscula T	54	T
+	Signo de suma	2B	+	U	Letra mayúscula U	55	U
´	Coma	2C	´	V	Letra mayúscula V	56	V
-	Guion/símbolo de menos	2D	-	W	Letra mayúscula W	57	W
•	Punto final	2E	•	X	Letra mayúscula X	58	X
/	Barra	2F	%2F	Y	Letra mayúscula Y	59	Y
0	Dígito cero	30	0	Z	Letra Z mayúscula	5A	Z
1	Dígito uno	31	1		Guión bajo	5F	__
2	Dígito dos	32	2	a	Letra minúscula a	61	a
3	Dígito tres	33	3	b	Letra minúscula b	62	b

Símbolo gráfico	Nombre	Valor hexadecimal	Forma de URI	Símbolo gráfico	Nombre	Valor hexadecimal	Forma de URI
4	Dígito cuatro	34	4	c	Letra minúscula c	63	c
5	Dígito cinco	35	5	d	Letra minúscula d	64	d
6	Dígito seis	36	6	e	Letra minúscula e	65	e
7	Dígito siete	37	7	f	Letra minúscula f	66	f
8	Dígito ocho	38	8	g	Letra minúscula g	67	g
9	Dígito nueve	39	9	h	Letra minúscula h	68	h
:	Dos puntos	3A		i	Letra minúscula i	69	i
;	Punto y coma	3B	;	j	Letra minúscula j	6A	j
<	Signo de menor que	3C	%3C	k	Letra minúscula k	6B	k
=	Signo de igual	3D	=	l	Letra minúscula l	6C	l
>	Signo de mayor que	3E	%3E	m	Letra minúscula m	6D	m
?	Signo de interrogación	3F	%3F	n	Letra minúscula n	6E	n
A	Letra mayúscula A	41	A	o	Letra minúscula o	6F	o
B	Letra mayúscula B	42	B	P	Letra minúscula p	70	P
C	Letra mayúscula C	43	C	q	Letra minúscula q	71	q
D	Letra mayúscula D	44	D	r	Letra minúscula r	72	r
E	Letra mayúscula E	45	E	s	Letra minúscula s	73	s
F	Letra mayúscula F	46	F	t	Letra minúscula t	74	t
G	Letra mayúscula G	47	G	u	Letra minúscula u	75	u
H	Letra mayúscula H	48	H	v	Letra minúscula v	76	v
I	Letra mayúscula I	49	I	w	Letra minúscula w	77	w
J	Letra mayúscula J	4A	J	x	Letra minúscula x	78	x
K	Letra mayúscula K	4B	K	y	Letra minúscula y	79	y
L	Letra mayúscula L	4C	L	z	Letra minúscula z	7A	z

B Glosario (no normativo)

Consulte *www. gs1.org/glossary* para obtener la versión más reciente del glosario.

Término	Lugar donde aparece la definición	Significado
Identificador de aplicación (AI)	[GS1GS]	Código numérico que identifica un elemento de datos en una cadena de elementos GS1.
Bits de atributo	Sección *11*	Un campo de 8 bits de información de control que se almacena en el banco de memoria de EPC de una etiqueta RFID Gen 2 cuando la etiqueta contiene un EPC. Los bits de atributo incluyen datos que guían el manejo del objeto al que se fija la etiqueta, por ejemplo un bit que indica la presencia de material peligroso.
Código de barras		Un portador de datos que contiene datos de texto en forma de marcas claras y oscuras que un lector óptico puede leer.
Información de control	Sección *9.1*	Información que utilizan las aplicaciones de captura de datos para ayudar a controlar el proceso de interacción con las etiquetas RFID. La información de control incluye datos que ayudan a una aplicación de captura a filtrar etiquetas de grandes poblaciones para aumentar la eficiencia de lectura, la información de manejo especial que afecta el comportamiento de la aplicación de captura, la información que controla las características de seguridad de las etiquetas, etc. La información de control generalmente *no* se transmite directamente a las aplicaciones comerciales, aunque puede influir en la forma en que una aplicación de captura presenta los datos comerciales al nivel de la aplicación comercial. A diferencia de los datos comerciales, la información de control no tiene equivalente en códigos de barras u otros portadores de datos.
Portador de datos		Término genérico para una marca o un dispositivo que se usa para incluir físicamente datos en un objeto físico. Algunos ejemplos de portadores de datos son los códigos de barras y las etiquetas RFID.
Código electrónico de producto (EPC)	Sección *4*	Identificador universal para un objeto físico. El EPC está diseñado para que a cada objeto físico de interés para los sistemas de información se le pueda asignar un EPC que sea único a nivel mundial y constante con el paso del tiempo. La representación primaria de un EPC es un URI de EPC de identidad pura (*q.v.*), que es una cadena única que se puede usar en sistemas de información, mensajes electrónicos, bases de datos y otros contextos. Una representación secundaria, la codificación binaria de EPC (*q.v.*) está disponible en las etiquetas RFID y en otros contextos en los que se requiere una representación binaria compacta.
EPC	Sección *4*	Consulte Código electrónico de producto
Banco de EPC (de una etiqueta RFID Gen 2)	[UHFC1G2]	Banco 01 de una etiqueta RFID Gen 2 según se especifica en [UHFC1G2]. El banco de EPC admite la codificación binaria de EPC de un EPC, junto con la información de control adicional, según se especifica en la Sección *7.9*.
Codificación binaria de EPC	Sección *13*	Una codificación compacta de un código electrónico de producto, junto con un valor de filtro (si el esquema de codificación incluye un valor de filtro), en una cadena de bits binaria que es adecuada para almacenarse en etiquetas RFID, incluido el banco de memoria de EPC de una etiqueta RFID Gen 2. Debido a los intercambios entre la capacidad de datos y la cantidad de bits en el valor codificado, existe más de un esquema de codificación binaria para determinados esquemas de EPC.
Esquema de codificación binaria de EPC	Sección *13*	Un formato particular para la codificación de un código electrónico de producto, junto con un valor de filtro en algunos casos, en una codificación binaria de EPC. Cada elemento de EPC tiene por lo menos un esquema de codificación binaria de EPC de una combinación especificada de elementos de datos. Debido a los intercambios entre la capacidad de datos y la cantidad de bits en el valor codificado, existe más de un esquema de codificación binaria para determinados esquemas de EPC. Una codificación binaria de EPC comienza con un encabezado de 8 bits que identifica el esquema de codificación binaria usado para esa codificación binaria; sirve para identificar cómo se debe interpretar el resto de la codificación binaria.
URI de identidad pura de EPC	Sección *6*	Ver URI de identidad pura de EPC.
URI de datos sin procesar de EPC	Sección *12*	Una representación del contenido completo del banco de memoria de EPC de una etiqueta RFID Gen 2.

Término	Lugar donde aparece la definición	Significado
Esquema de EPC	Sección 6	Formato particular para la construcción de un código electrónico de producto a partir de una combinación especificada de elementos de datos. Un URI de identidad pura comienza con el nombre del esquema de EPC usado para ese URI, que sirve para garantizar que el URI es único a nivel mundial y para identificar cómo se interpreta el resto del URI. Cada tipo de clave GS1 tiene un esquema de EPC correspondiente que permite la construcción de un EPC que corresponde al valor de una llave GS1, en determinadas condiciones. Existen otros esquemas de EPC que permiten la construcción de EPC no relacionados con las llaves GS1.
URI de la etiqueta EPC	Sección 12	Representación del contenido completo del banco de memoria de EPC de una etiqueta RFID Gen 2, en forma de un identificador de recursos uniforme de Internet que incluye una representación decodificada de campos de datos de EPC, que se puede usar cuando el banco de memoria de EPC contiene una codificación binaria de EPC válida. Dado que el URI de etiqueta de EPC representa el contenido completo del banco de memoria de EPC, incluye información de control adicional al EPC, en contraste con el URI de EPC de identidad pura.
Identificación de etiqueta extendida (XTID)	Sección 16	Información que se puede incluir en el banco de TID de una etiqueta RFID Gen 2 además del fabricante y la información del modelo. La XTID puede incluir un número de serie único asignado por el fabricante y también otros datos que describen las capacidades de la etiqueta.
Valor de filtro	Sección 10	Un campo de 3 bits de información de control que se almacena en el banco de memoria de EPC de una etiqueta RFID Gen 2 cuando la etiqueta contiene un determinado tipo de EPC. El valor de filtro facilita la lectura de las etiquetas RFID deseadas en un entorno en el que pueden estar presentes otras etiquetas, como la lectura de una etiqueta de tarima en presencia de una gran cantidad de etiquetas para artículos.
Etiqueta RFID Gen 2	Sección 7.9	Una etiqueta RFID que cumple con uno de los protocolos de interfaz aérea de la familia EPCglobal Gen 2. Esto incluye la interfaz aire Gen 2 clase 1 de UHF [UHFC1G2], y otros estándares que actualmente se están desarrollando en EPCglobal.
Prefijo GS1 de empresa	[GS1GS]	Parte del número de identificación del sistema GS1 que consta de un prefijo GS1 y de un número de empresa; ambos son asignados por las organizaciones miembro de GS1.
Cadena de elementos GS1	[GS1GS]	La combinación de un identificador de aplicación GS1 y un campo de datos de identificador de aplicación GS1.
Clave GS1	[GS1GS]	Término genérico para las claves de identificación definidas en las Especificaciones Generales de GS1 [GS1GS], como GTIN, SSCC, GLN, GRAI, GIAI, GSRN, GDTI, GS1N, GINC, CPID, GCN y GMN.
URI de identidad pura EPC	Sección 6	La representación concreta principal de un código electrónico de producto. El URI de EPC de identidad pura es un identificador de recursos uniforme de Internet que contiene un código electrónico de producto y ningún otro dato.
Etiqueta de identificación por radio frecuencia (RFID)		Portador de datos que contiene datos binarios, que se pueden fijar a un objeto físico, y que transmite los datos a un dispositivo de interrogación ("lector") a través de ondas de radio.
Banco reservado (de una etiqueta RFID Gen 2)	[UHFC1G2]	Banco 00 de una etiqueta RFID Gen 2 según se especifica en [UHFC1G2]. El banco reservado contiene la contraseña de acceso y la contraseña de destrucción.
Identificación de etiqueta (TID)	[UHFC1G2]	Información que describe una etiqueta RFID Gen 2, en contraposición al objeto físico al que está adherida la etiqueta. La TID incluye una indicación del fabricante y del modelo de la etiqueta, y puede incluir información del TID extendido (XTID).
Banco de TID (de una etiqueta RFID Gen 2)	[UHFC1G2]	Banco 10 de una etiqueta RFID Gen 2 según se especifica en [UHFC1G2]. El banco de TID contiene la TID y la XTID *(q.v.)*.
Identificador de recursos uniforme (URI)	[RFC3986]	Secuencia compacta de caracteres que identifica un resumen o un recurso físico. Un URI se puede clasificar como un nombre de recurso uniforme (URN) o como un localizador de recursos uniforme (URL) *q.v.*.
Localizador de recursos uniformes (URL)	[RFC3986]	Un identificador de recursos uniforme (URI) que, además de identificar un recurso, proporciona los medios para localizar el recurso mediante la descripción de su mecanismo de acceso principal (por ejemplo, la "ubicación" de su red).

Término	Lugar donde aparece la definición	Significado
Nombre de recurso uniforme (URN)	[RFC3986], [RFC2141]	Un identificador de recursos uniforme (URI) que es parte del esquema urn especificado en [RFC2141]. El URI se refiere a un recurso específico independiente de la ubicación de su red u otro método de acceso, o que posiblemente no tenga ninguna ubicación de red. El término URN también puede referirse a cualquier otro URI con propiedades similares. Dado que un código electrónico de producto es un identificador único para un objeto físico que no necesariamente tiene una ubicación de red u otro método de acceso, los URN se usan para representar EPC.
Banco de memoria del usuario (de una etiqueta RFID Gen 2)	[UHFC1G2]	Banco 11 de una etiqueta RFID Gen 2 según se especifica en [UHFC1G2]. La memoria del usuario se puede usar para almacenar elementos de datos empresariales adicionales al EPC.

C Referencias

[ASN.1] CCITT, "Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)", Recomendación de CCITT X.209, enero de 1988.

[EPCAF] F. Armenio et al, "EPCglobal Architecture Framework", versión 1.7, mayo de 2015, https://www.gs1.org/id-keys-epcrfid-epcis/epc-rfid-architecture-framework/1-6

[GS1GS19.1] "Especificaciones Generales de GS1", versión 19.1, julio de 2019, publicadas por GS1, Blue Tower, Avenue Louise 326, bte10, Bruselas 1009, B-1050, Bélgica, www.gs1.org.

[ISO15961] ISO/IEC, "Information technology - Radio frequency identification (RFID) for item management - Data protocol: application interface", ISO/IEC 15961:2004, octubre de 2004.

[ISO15962] ISO/IEC, "Information technology - Radio frequency identification (RFID) for item management - Data protocol: data encoding rules and logical memory functions", ISO/IEC 15962:2004, octubre de 2004. (Cuando se publique la 2ª edición de ISO/IEC 15962, debería usarse en lugar de la versión anterior. Las referencias aquí incluidas al Anexo D de [15962] se refieren únicamente a la 2ª edición de ISO/IEC 15962, o a una edición posterior.)

[ISODir2] ISO, "Rules for the structure and drafting of International Standards (Directivas ISO/IEC, parte 2, 2001, 4a. Edición)", julio de 2002.

[RFC2141] R. Moats, "URN Syntax," RFC2141, mayo de 1997, http://www.ietf.org/rfc/rfc2141.

[RFC3986] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax," RFC3986, enero de 2005, http://www.ietf.org/rfc/rfc3986.

[ONS1.0.1] EPCglobal, "EPCglobal Object Naming Service (ONS), Versión 1.0.1", estándar ratificado por EPCglobal, mayo de 2008, http://www.epcglobalinc.org/standards/ons/ons_1_0_1-standard-20080529.pdf.

[SPEC2000] Air Transport Association, "Spec 2000 E-Business Specification for Materials Management," May 2009, http://www.spec2000.com.

[UHFC1G2] EPCglobal, "EPC™ Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz - 960 MHz Versión 1.2.0," EPCglobal Specification, May 2008, http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_2_0-standard-20080511.pdf.

[UID] "United States Department of Defense Guide to Uniquely Identifying Items" v2.0 (1st October 2008), http://www.acq.osd.mil/dpap/UID/attachments/DoDUIDGuide.pdf

[USDOD] "United States Department of Defense Suppliers' Passive RFID Information Guide," http://www.dodrfid.org/supplierguide.htm

D Vectores de bits ampliables

Un vector de bits ampliable (EBV) es una estructura de datos con un rango de datos ampliable.

Un EBV es una matriz de bloques. Cada bloque contiene un bit ampliable único seguido de una cantidad específica de bits de datos. Si B es la cantidad total de bits en un bloque, entonces un bloque contiene bits de datos B - 1. El EBV-***n*** de la notación que se utilizó en esta especificación indica un EBV con un tamaño de bloque de *n*; por ejemplo, EBV- 8 denota un EBV con B=8.

El valor de los datos representados por un EBV es simplemente la secuencia de bits formada por los bits de datos, leídos de izquierda a derecha, ignorando todos los bits ampliables. El último bloque de un EBV cuenta con un bit ampliable de cero, y todos los bloques de un EBV que preceden al último bloque (si lo hay) tienen un bit ampliable de uno.

La tabla siguiente ilustra diversos valores representados en los formatos EBV-6 y EBV-8. Se agregan espacios a los EBV para claridad visual.

Valor	EBV-6	EBV-8
0	000000	00000000
1	000001	00000001
31 (2^5-1)	011111	00011111
32 (2^5)	100001 000000	00100000
33 (2^5+1)	100001 000001	00100001
127 (2^7-1)	100011 011111	01111111
128(2^7)	100100 000000	10000001 00000000
129 (2^7+1)	100100 000001	10000001 00000001
16384 (2^{14})	000000 100000 110000	00000000 10000000 10000001

La especificación de objetos empaquetados que se encuentra en [7] utiliza EBV-3, EBV-6, y EBV-8.

E Ejemplos (no normativos): codificación y decodificación de un EPC

Esta sección representa dos ejemplos detallados que muestran la codificación y la decodificación entre el número de identificación global seriado (SGTIN) y el banco de memoria del EPC de la etiqueta RFID Gen 2, así como ejemplos del resumen que muestran diversas codificaciones de todos los esquemas de EPC.

Como estos son únicamente ejemplos ilustrativos, en todos los casos, se deben consultar las secciones normativas indicadas de esta especificación con el fin de conocer las normas definitivas para codificar y decodificar. Los diagramas y las notas adjuntas en esta sección no tienen como finalidad ser una especificación completa para la codificación o decodificación; sino que sirven únicamente para ilustrar lo más destacado sobre cómo funcionan los procedimientos normativos de codificación y decodificación. Los procedimientos para codificar otros tipos de identificadores son diferentes de maneras significativas, y se deben consultar las secciones adecuadas de esta especificación.

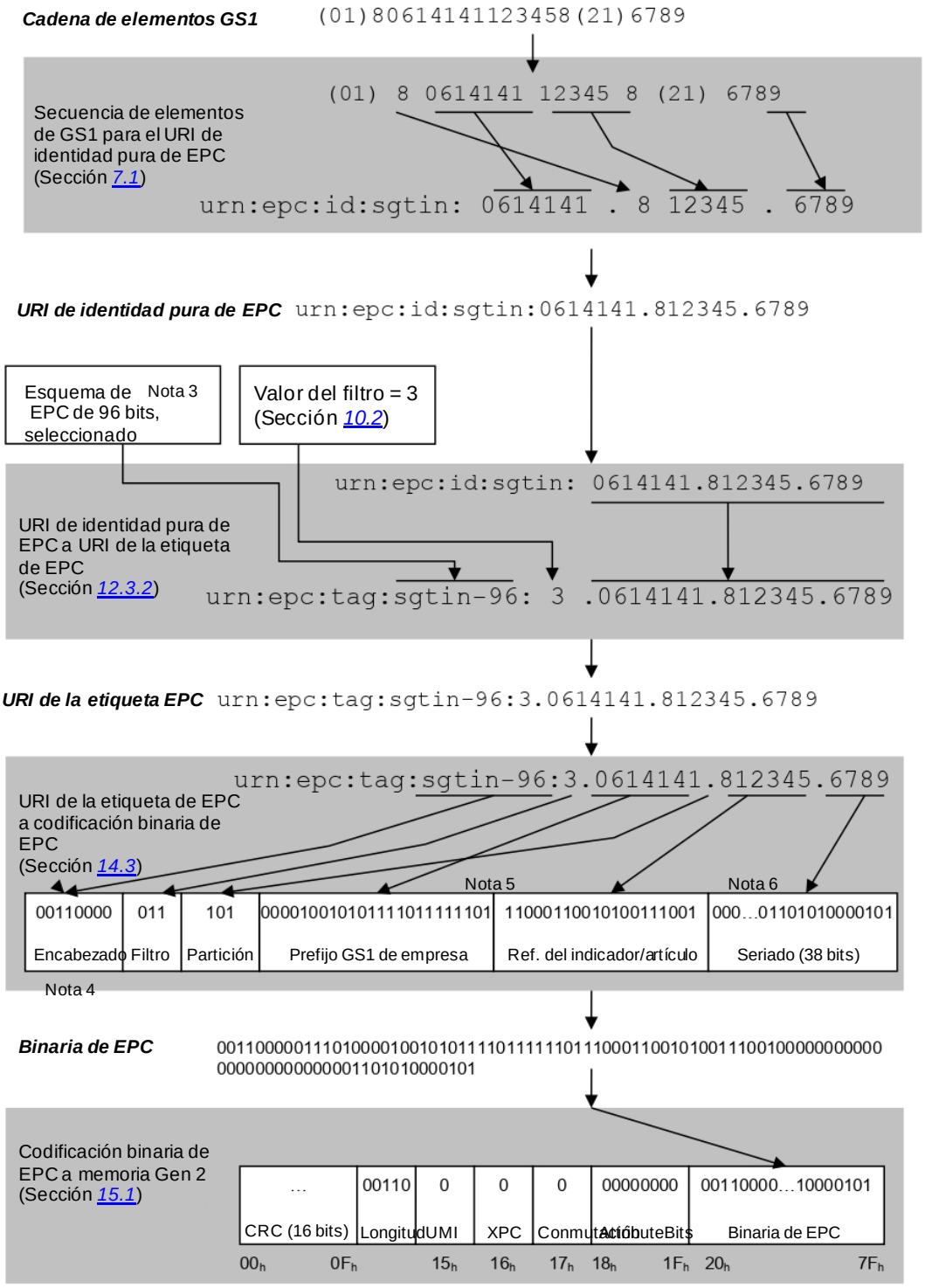
E.1 Codificación de un número global de artículo comercial seriado (SGTIN) a SGTIN-96

Este ejemplo ilustra la codificación de una secuencia de elementos GS1 que contiene un número global de artículo comercial seriado (SGTIN) en la etiqueta RFID Gen 2 de EPC, utilizando el esquema de EPC SGTIN-96, con pasos intermedios que incluyen el URI de EPC, el URI de etiqueta de EPC, y la codificación binaria de EPC.

En algunas aplicaciones, únicamente una parte de esta ilustración es significante. Por ejemplo, es probable que una aplicación solo necesite transformar una secuencia de elementos de GS1 en un URI de EPC, en cuyo caso únicamente se necesita la parte superior de la ilustración.

La ilustración que se muestra a continuación hace referencia a las notas siguientes:

- **Nota 1**: El paso de convertir una secuencia de elementos de GS1 en un URI de identidad pura de EPC requiere que se determine la cantidad de dígitos en el Prefijo GS1 de empresa; por ejemplo, como referencia a una tabla externa de prefijos de empresa. En este ejemplo se muestra que el Prefijo GS1 de empresa tiene siete dígitos.
- **Nota 2**: En el URI de identidad pura de EPC no se incluye el dígito de verificación en el GTIN, tal como aparece en la secuencia de elementos de GS1.
- **Nota 3**: El esquema de EPC SGTIN-96 únicamente se puede utilizar si el número de serie cumple con ciertas restricciones. Específicamente, el número de serie solo debe (a) constar de caracteres de dígitos; (b) no debe comenzar con un dígito cero (a menos que todo el número de serie sea el dígito único "0"); y (c) corresponde a un número decimal cuyo valor numérico es menor que 2^{38} (menor que 274,877,906,944). Para todos los números de serie, se debe utilizar el esquema de EPC SGTIN-198. Cabe mencionar que el URI de EPC es idéntico, independientemente si se utiliza SGTIN-96 o SGTIN-198 en la etiqueta de RFID.
- **Nota 4**: Los valores de encabezado de codificación binaria de EPC se codificarán en la sección *14.2*.
- **Nota 5**: La cantidad de bits en el Prefijo GS1 de empresa y en los campos de referencia del artículo/indicador en la codificación binaria de EPC depende de una cantidad de dígitos en la porción del Prefijo GS1 de empresa del URI de EPC, y esto se indica por medio de un código en el campo Partición de la codificación binaria de EPC. Consulte la sección *14.2*. (Únicamente para el EPC de SGTIN).
- **Nota 6**: El campo seriado de la codificación binaria de EPC para SGTIN-96 es 38 bits; sin embargo, no se muestran todos los bits debido a las limitantes de espacio.


Cadena de elementos GS1
(01)80614141123458(21)6789
Secuencia de elementos de GS1 para el URI de identidad pura de EPC (Sección 7.1)
(01) 8 0614141 12345 8 (21) 6789
urn:epc:id:sgtin: 0614141 . 8 12345 . 6789
URI de identidad pura de EPC
urn:epc:id:sgtin:0614141.812345.6789
Esquema de EPC de 96 bits, seleccionado
Nota 3
Valor del filtro = 3 (Sección 10.2)
URI de identidad pura de EPC a URI de la etiqueta de EPC (Sección 12.3.2)
urn:epc:id:sgtin: 0614141.812345.6789
urn:epc:tag:sgtin-96: 3 .0614141.812345.6789
URI de la etiqueta de EPC
urn:epc:tag:sgtin-96:3.0614141.812345.6789
URI de la etiqueta de EPC a codificación binaria de EPC (Sección 14.3)
Nota 5
Nota 6
00110000
011
101
00001001010111101111101
110001100101001110 01
000…01101010000101
Encabezado
Filtro
Partición
Prefijo GS1 de empresa
Ref. del indicador/artículo
Seriado (38 bits)
Nota 4
Binaria de EPC
00110000011101000010010101111011111101110001100101001110010000000000
00000000000000011010100001 01
Codificación binaria de EPC a memoria Gen 2 (Sección 15.1)
…
00110
0
0
0
00000000
00110000…10000101
CRC (16 bits)
Longitud
UMI
XPC
Conmutación
AttributeBits
Binaria de EPC
00h
0Fh
15h
16h
17h
18h
1Fh
20h
7Fh

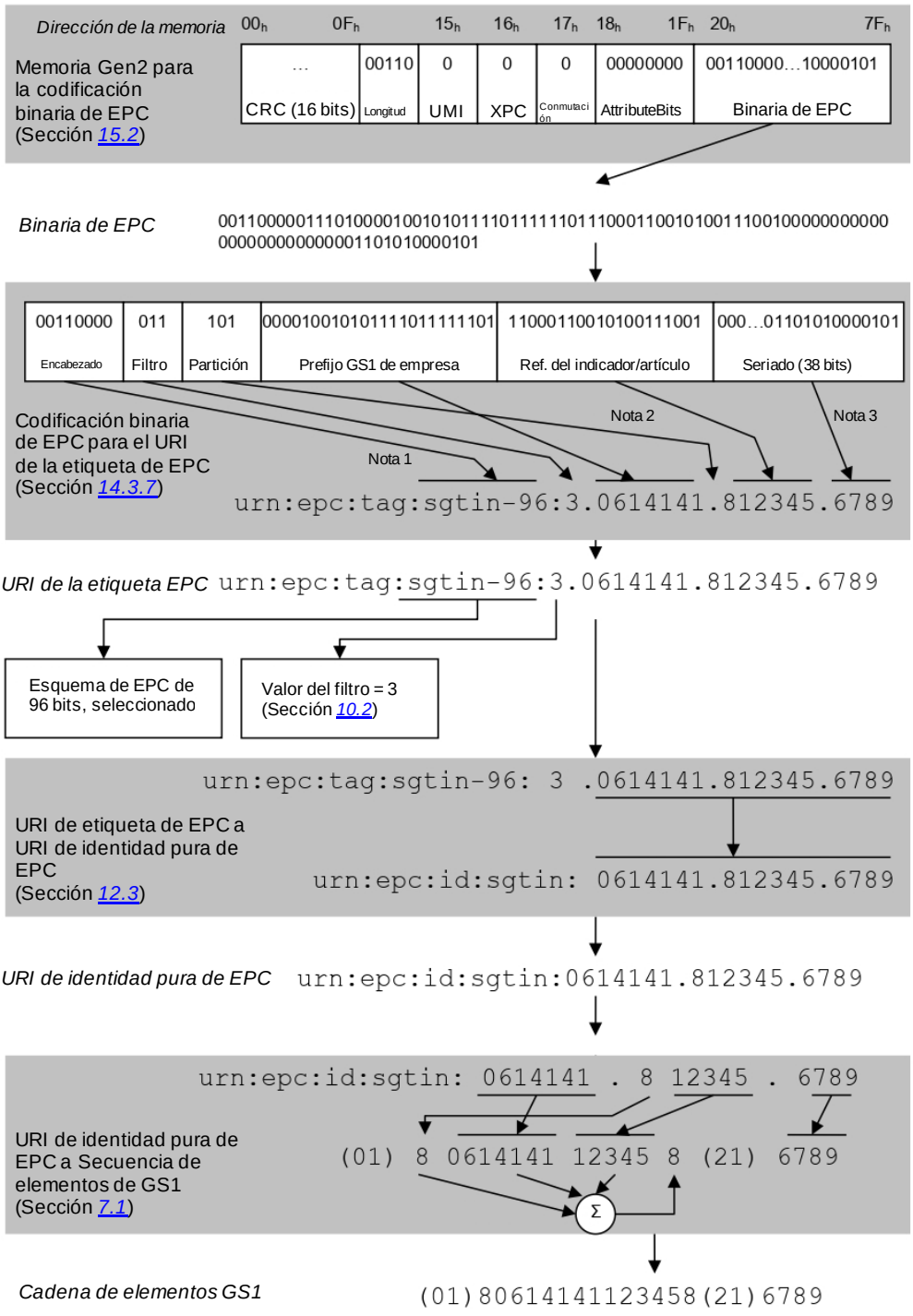
E.2 Codificación de un SGTIN-96 para un Número global de artículo comercial seriado (SGTIN)

Este ejemplo ilustra la decodificación de una etiqueta de RFID Gen 2 de EPC que contiene una codificación binaria de EPC SGTIN-96 en una secuencia de elementos de GS1 que contiene un número global de artículo comercial seriado (SGTIN), con pasos intermedios que incluyen la codificación binaria de EPC, el URI de la etiqueta de EPC, y el URI de EPC.

En algunas aplicaciones, únicamente una parte de esta ilustración es significante. Por ejemplo, es probable que una aplicación solo necesite convertir una codificación binaria de EPC en un URI de EPC, en cuyo caso únicamente se necesita la parte superior de la ilustración.

La ilustración que se muestra a continuación hace referencia a las notas siguientes:

- **Nota 1**: El encabezado de la codificación binaria de EPC indica cómo interpretar el recordatorio de los datos binarios, y el nombre del esquema de EPC que se incluirá en el URI de la etiqueta de EPC. Los valores de encabezado de codificación binaria de EPC se codificarán en la sección *14.2*.
- **Nota 2**: El campo de Partición de la codificación binaria de EPC contiene un código que indica la cantidad de bits en el campo de Prefijo GS1 de empresa y el campo de referencia del artículo/indicador. Además, el código de la división determina la cantidad de dígitos decimales que se utilizarán para dichos campos en el URI de la etiqueta de EPC (la representación decimal para esos dos campos se rellena a la izquierda con caracteres cero, si es necesario). Consulte la Sección *14.2*. (únicamente para el EPC de SGTIN).
- **Nota 3**: Para el esquema de EPC SGTIN-96, el campo de Número de serie se decodifica al interpretar los bits como un número entero binario, y al convertirlo a un número decimal sin ceros iniciales (a menos que todos los bits de número de serie sean cero, lo que se decodifica como la secuencia “0”). Los números de serie que contienen caracteres sin dígitos o que comienzan con caracteres de ceros iniciales pueden codificarse únicamente en el esquema de EPC SGTIN-198.
- **Nota 4**: El dígito de verificación en la secuencia de elementos de GS1 se calcula a partir de otros dígitos en el URI de identidad pura de EPC, tal como se especifica en la Sección *7.1*.


Dirección de la memoria
00h 0Fh 15h 16h 17h 18h 1Fh 20h 7Fh
Memoria Gen2 para la codificación binaria de EPC (Sección 15.2)
... CRC (16 bits)
00110 Longitud
0 UMI
0 XPC
0 Conmutación
00000000 AttributeBits
00110000...10000101 Binaria de EPC
Binaria de EPC
00110000011101000010010101111011111101110001100101001110010000000000000000000000000011010100000101
00110000 Encabezado
011 Filtro
101 Partición
000010010101111011111101 Prefijo GS1 de empresa
110001100101001110 01 Ref. del indicador/artículo
000...01101010000101 Seriado (38 bits)
Codificación binaria de EPC para el URI de la etiqueta de EPC (Sección 14.3.7)
Nota 1
Nota 2
Nota 3
urn:epc:tag:sgtin-96:3.0614141.812345.6789
URI de la etiqueta EPC urn:epc:tag:sgtin-96:3.0614141.812345.6789
Esquema de EPC de 96 bits, seleccionado
Valor del filtro = 3 (Sección 10.2)
urn:epc:tag:sgtin-96: 3 .0614141.812345.6789
URI de etiqueta de EPC a URI de identidad pura de EPC (Sección 12.3)
urn:epc:id:sgtin: 0614141.812345.6789
URI de identidad pura de EPC urn:epc:id:sgtin:0614141.812345.6789
urn:epc:id:sgtin: 0614141 . 8 12345 . 6789
URI de identidad pura de EPC a Secuencia de elementos de GS1 (Sección 7.1)
(01) 8 0614141 12345 8 (21) 6789
Σ
Cadena de elementos GS1
(01)80614141123458(21)6789

E.3 Ejemplos del resumen de todos los esquemas de EPC

En todos los ejemplos siguientes, se asume que el Prefijo GS1 de empresa 0614141 tiene siete dígitos.

SGTIN-96	
Cadena de elementos GS1	(01)80614141123458(21)6789
URI de EPC	urn:epc:id:sgtin:0614141.812345.6789
URI de la etiqueta EPC	urn:epc:tag:sgtin-96:3.0614141.812345.6789
Codificación binaria de EPC (hex)	3074257BF7194E4000001A85

SGTIN-198	
Cadena de elementos GS1	(01)70614141123451(21)32a/b
URI de EPC	urn:epc:id:sgtin:0614141.712345.32a%2Fb
URI de la etiqueta EPC	urn:epc:tag:sgtin-198:3.0614141.712345.32a%2Fb
Codificación binaria de EPC (hex)	3674257BF6B7A659B2C2BF10000000000000000000000000000

SSCC-96	
Cadena de elementos GS1	(00) 106141412345678908
URI de EPC	urn:epc:id:sscc:0614141.1234567890
URI de la etiqueta EPC	urn:epc:tag:sscc-96:3.0614141.1234567890
Codificación binaria de EPC (hex)	3174257BF4499602D2000000

SGLN-96	
Cadena de elementos GS1	(414) 0614141123452 (254) 5678
URI de EPC	urn:epc:id:sgln:0614141.12345.5678
URI de la etiqueta EPC	urn:epc:tag:sgln-96:3.0614141.12345.5678
Codificación binaria de EPC (hex)	3274257BF46072000000162E

SGLN-195	
Cadena de elementos GS1	(414) 0614141123452 (254) 32a/b
URI de EPC	urn:epc:id:sgln:0614141.12345.32a%2Fb
URI de la etiqueta EPC	urn:epc:tag:sgln-195:3.0614141.12345.32a%2Fb
Codificación binaria de EPC (hex)	3974257BF46072CD9615F880000000000000000000000000

GRAI-96	
Cadena de elementos GS1	(8003) 0061414112345256 78
URI de EPC	urn:epc:id:grai:0614141.12345.5678
URI de la etiqueta EPC	urn:epc:tag:grai-96:3.0614141.12345.5678
Codificación binaria de EPC (hex)	3374257BF40C0E400000162E

GRAI-170	
Cadena de elementos GS1	(8003) 00614141123452 32a/b

GRAI-170	
URI de EPC	urn:epc:id:grai:0614141.12345.32a%2Fb
URI de la etiqueta EPC	urn:epc:tag:grai-170:3.0614141.12345.32a%2Fb
Codificación binaria de EPC (hex)	3774257BF40C0E59B2C2BF100000000000000000000

GIAI-96	
Cadena de elementos GS1	(8004) 06141415678
URI de EPC	urn:epc:id:giai:0614141.5678
URI de la etiqueta EPC	urn:epc:tag:giai-96:3.0614141.5678
Codificación binaria de EPC (hex)	3474257BF40000000000162E

GIAI-202	
Cadena de elementos GS1	(8004) 061414132a/b
URI de EPC	urn:epc:id:giai:0614141.32a%2Fb
URI de la etiqueta EPC	urn:epc:tag:giai-202:3.0614141.32a%2Fb
Codificación binaria de EPC (hex)	3874257BF59B2C2BF10000000000000000000000000000000000

GSRN-96	
Cadena de elementos GS1	(8018) 061414112345678902
URI de EPC	urn:epc:id:gsrn:0614141.1234567890
URI de la etiqueta EPC	urn:epc:tag:gsrn-96:3.0614141.1234567890
Codificación binaria de EPC (hex)	2D74257BF4499602D2000000

GSRNP-96	
Cadena de elementos GS1	(8017) 061414112345678902
URI de EPC	urn:epc:id:gsrnp:0614141.1234567890
URI de la etiqueta EPC	urn:epc:tag:gsrnp-96:3.0614141.1234567890
Codificación binaria de EPC (hex)	2E74257BF4499602D2000000

GDTI-96	
Cadena de elementos GS1	(253) 06141411234525678
URI de EPC	urn:epc:id:gdti:0614141.12345.5678
URI de la etiqueta EPC	urn:epc:tag:gdti-96:3.0614141.12345.5678
Codificación binaria de EPC (hex)	2C74257BF46072000000162E

GDTI-174	
Cadena de elementos GS1	(253) 4012345987652ABCDefgh012345678
URI de EPC	urn:epc:id:gdti:4012345.98765.ABCDefgh01234567 8
URI de la etiqueta EPC	urn:epc:tag:gdti- 174:3.4012345.98765.ABCDefgh012345678
Codificación binaria de EPC (hex)	3E74F4E4E7039B061438997367D0C18B266D1AB66EE0

CPI-96	
Cadena de elementos GS1	(8010) 061414198765 (8011) 12345
URI de EPC	urn:epc:id:cpi:0614141.98765.12345
URI de la etiqueta EPC	urn:epc:tag:cpi-96:3.0614141.98765.12345
Codificación binaria de EPC (hex)	3C74257BF400C0E680003039

CPI-var	
Cadena de elementos GS1	(8010) 06141415PQ7/Z43 (8011) 12345
URI de EPC	urn:epc:id:cpi:0614141.5PQ7%2FZ43.12345
URI de la etiqueta EPC	urn:epc:tag:cpi-var:3.0614141.5PQ7%2FZ43.12345
Codificación binaria de EPC (hex)	3D74257BF75411DEF6B4CC00000003039

SGCN-96	
Cadena de elementos GS1	(255) 401234567890104711
URI de EPC	urn:epc:id:sgcn:4012345.67890.04711
URI de la etiqueta EPC	urn:epc:tag:sgcn-96:3.4012345.67890.04711
Codificación binaria de EPC (hex)	3F74F4E4E612640000019907

GID-96	
URI de EPC	urn:epc:id:gid:31415.271828.1414
URI de la etiqueta EPC	urn:epc:tag:gid-96:31415.271828.1414
Codificación binaria de EPC (hex)	350007AB70425D4000000586

USDOD-96	
URI de EPC	urn:epc:id:usdod:CAGEY.5678
URI de la etiqueta EPC	urn:epc:tag:usdod-96:3.CAGEY.5678
Codificación binaria de EPC (hex)	2F32043414745590000162E

ADI-var	
URI de EPC	urn:epc:id:adi:35962.PQ7VZ4.M37GXB92
URI de la etiqueta EPC	urn:epc:tag:adi-var:3.35962.PQ7VZ4.M37GXB92
Codificación binaria de EPC (hex)	3B0E0CF5E76C9047759AD00373DC7602E7200

ITIP-110	
Cadena de elementos GS1	(8006) 040123451234560102 (21) 981
URI de EPC	urn:epc:id:itip:4012345.012345.01.02.981
URI de la etiqueta EPC	urn:epc:tag:itip-110:0.4012345.012345.01.02.981
Codificación binaria de EPC (hex)	4014F4E4E40C0E40820000000F54

ITIP-212	
Cadena de elementos GS1	(8006) 040123451234560102 (21) mw133

ITIP-212	
URI de EPC	urn:epc:id:itip:4012345.012345.01.02.mw133
URI de la etiqueta EPC	urn:epc:tag:itip-212:0.4012345.012345.01.02.mw133
Codificación binaria de EPC (hex)	4114F4E4E40C0E4082DBDD8B3660000000000000000000000000000000

F Tabla de los ID de los objetos empaquetados para el formato de datos 9

Esta sección proporciona la tabla de los ID de los objetos empaquetados para el formato de datos 9, que define los valores de los ID de los objetos empaquetados, OID, y las secuencias del formato para los identificadores de aplicación GS1.

La sección F.1 es una lista no normativa del contenido de la tabla de los ID para el formato de datos 9, en un formato tabular, legible para el ser humano. La sección F.2 es la tabla normativa, legible para máquina, en formato de valores separados por comas, tal como se registra con la ISO.

F.1 Formato tabular (no normativo)

Esta sección es una lista no normativa del contenido de la tabla de los ID para el formato de datos 9, en un formato tabular, legible para el ser humano. Consulte la Sección F.2 para ver la tabla normativa, legible para máquina, en formato de valores separados por comas, tal como se registra con la ISO.

K-Texto = Tabla de ID AI GS1 para ISO/IEC 15961 Formato 9						
K-Versión	= 1.00					
K-ISO15434=05						
K-Texto = Tabla base primaria						
K-TableID = F9B0						
K-RootOID = urn:oid :1.0.15961.9						
K-IDsize = 90						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
00	1	0	00	SSCC (Código Seriado de Contenedor de Envío)	SSCC	18n
01	2	1	01	Número global de artículo comercial	GTIN	14n
02 + 37	3	(2)(37)	(02)(37)	GTIN + Recuento de los artículos comerciales contenidos en una unidad comercial	CONTENIDO + RECUENTO	(14n)(1*8n)
10	4	10	10	Número de lote	LOTE	1*20an
11	5	11	11	Fecha de producción (AAMMDD)	FECHA DE PROD	6n
12	6	12	12	Fecha de vencimiento (AAMMDD)	FECHA DE VENCIMIENTO	6n
13	7	13	13	Fecha de empaque (AAMMDD)	Fecha de empaquetado	6n
15	8	15	15	Fecha de consumo preferente (AAMMDD)	FECHA DE CONSUMO PREFERENTE O FECHA LÍMITE DE VENTA	6n
17	9	17	17	Fecha de caducidad (AAMMDD)	EXPIRA O CADUCA	6n
20	10	20	20	Variante de producto interno	VARIANTE	2n
21	11	21	21	Número de serie	SERIADO	1*20an
22	12	22	22	Variante de producto de consumo	CPV	1*20an
240	13	240	240	Identificación adicional del producto asignada por el fabricante	IDENTIFICACIÓN ADICIONAL	1*30an
241	14	241	241	Número de pieza del cliente	NO. DE PARTE DEL CLIENTE	1*30an
242	15	242	242	Número de variación hecho a la medida	NÚMERO DE VARIACIÓN	1*6n
250	16	250	250	Número de serie secundario	DE SERIE SECUNDARIO	1*30an

K-Texto = Tabla de ID AI GS1 para ISO/IEC 15961 Formato 9						
251	17	251	251	Referencia a la entidad fuente	REF. A LA FUENTE	1*30an
253	18	253	253	Identificador global de tipo de documento	DOC. ID	13n 0*17an
30	19	30	30	Recuento de artículos variable (artículo comercial de medida variable)	VAR. CANTIDAD	1*8n
310n 320n, etc.	20	K-Secundario = S00		Peso neto, kilogramos o libras u onzas troy (artículo comercial de medida variable)		
311n 321n, etc.	21	K-Secundario = S01		Longitud de la primera dimensión (artículo comercial de medida variable)		
312n 324n, etc.	22	K-Secundario = S02		Ancho, diámetro o segunda dimensión (artículo comercial de medida variable)		
313n 327n, etc.	23	K-Secundario = S03		Profundidad, grosor, altura o tercera dimensión (artículo comercial de medida variable)		
314n 350n, etc.	24	K-Secundario = S04		Área (artículo comercial de medida variable)		
315n 316n, etc.	25	K-Secundario = S05		Volumen neto (artículo comercial de medida variable)		
330n o 340n	26	330%x30-36 / 340%x30-36	330%x30-36 / 340%x30-36	Peso logístico, kilogramos o libras	PESO BRUTO (kg) o (lb)	6n / 6n
331n, 341n, etc.	27	K-Secundario = S09		Longitud o primera dimensión		
332n, 344n, etc.	28	K-Secundario = S10		Ancho, diámetro o segunda dimensión		
333n, 347n, etc.	29	K-Secundario = S11		Profundidad, grosor, altura o tercera dimensión		
334n 353n, etc.	30	K-Secundario = S07		Área de logística		
335n 336n, etc.	31	K-Secundario = S06	335%x30-36	Volumen logístico		
337(***)	32	337%x30-36	337%x30-36	Kilogramos por metro cuadrado	KG PER m^2	6n
390n o 391n	33	390%x30-39 / 391%x30-39	390%x30-39 / 391%x30-39	Importe por pagar - zona monetaria única o con código de moneda ISO	CANTIDAD	1*15n / 4*18n

K-Texto = Tabla de ID AI GS1 para ISO/IEC 15961 Formato 9						
392n o 393n	34	392%x30-39 / 393%x30-39	392%x30-39 / 393%x30-39	Importe por pagar del artículo comercial de medida variable - unidad monetaria única o ISO cc	PRECIO	1*15n / 4*18n
400	35	400	400	Número de orden de compra del cliente	NÚMERO DE ORDEN	1*30an
401	36	401	401	Número global de identificación para consignación	GINC	1*30an
402	37	402	402	Número global de identificación de envío	GS1N	17n
403	38	403	403	Código de ruta	RUTA	1*30an
410	39	410	410	Enviar a - entregar al Número Global de Localización	ENVÍO A LOC	13n
411	40	411	411	Cobrar a - facturar al Número Global de Localización	FACTURAR A	13n
412	41	412	412	Comprado del Número Global de Localización	COMPRADO A	13n
413	42	413	413	Enviar a - entregar a - reenviar al Número Global de Localización	ENVÍO PARA LOC	13n
414 y 254	43	(414) [254]	(414) [254]	Identificación de una localización física GLN, y extensión opcional	No. LOC + EXTENSIÓN GLN	(13n) [1*20an]
415 y 8020	44	(415) (8020)	(415) (8020)	Número Global de Localización de la parte que factura y Número de referencia de talón de pago	PAGO + No. DE REF	(13n) (1*25an)
420 o 421	45	(420/421)	(420/421)	Enviar a – entregar al código postal	ENVÍO A DESTINO	(1*20an / 3n 1*9an)
422	46	422	422	País de origen de un artículo comercial	ORIGEN	3n
423	47	423	423	País de procesamiento inicial	PAÍS - PROCESAMIENTO INICIAL.	3*15n
424	48	424	424	País de procesamiento	PAÍS - PROCES.	3n
425	49	425	425	País de desensamble	PAÍS - DESENSAMBLE	3n
426	50	426	426	País que cubre la cadena de proceso completa	PAÍS - PROCESO COMPLETO	3n
7001	51	7001	7001	Número de stock de la OTAN	NSN	13n
7002	52	7002	7002	Clasificación de las carcasas y cortes de carne de ONU/CEPE	CORTE CARNE	1*30an
7003	53	7003	7003	Fecha y hora de caducidad	FECHA/HORA DE CADUCIDAD	10n
7004	54	7004	7004	Potencia activa	POTENCIA ACTIVA	1*4n
703s	55	7030	7030	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	56	7031	7031	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an

K-Texto = Tabla de ID AI GS1 para ISO/IEC 15961 Formato 9						
703s	57	7032	7032	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	58	7033	7033	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	59	7034	7034	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	60	7035	7035	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	61	7036	7036	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	62	7037	7037	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	63	7038	7038	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
703s	64	7039	7039	Número de aprobación del procesador con código ISO del país	PROCESADOR # s	3n 1*27an
8001	65	8001	8001	Productos en rollo - ancho, largo, diámetro interno, dirección y empalmes	DIMENSIONES	14n
8002	66	8002	8002	Identificador electrónico seriado para teléfonos celulares	N.° CMT	1*20an
8003	67	8003	8003	Identificador global de activos retornables	GRAI	14n 0*16an
8004	68	8004	8004	Identificador global individual de activo	GIAI	1*30an
8005	69	8005	8005	Precio por unidad de medida	PRECIO POR UNIDAD	6n
8006	70	8006	8006	Identificación del componente de un artículo comercial	ITIP	18n
8007	71	8007	8007	Número de cuenta bancaria internacional	IBAN	1*34an
8008	72	8008	8008	Fecha y hora de producción	HORA PROD	8*12n
8018	73	8018	8018	Número global de relación del servicio - Receptor	GSRN - RECEPTOR	18n
8100 8101, etc.	74	K-Secundario = S08		Código de cupón		
90	75	90	90	Información acordada mutuamente entre socios comerciales (incluidos los FACT DI)	INTERNO	1*30an
91	76	91	91	Información interna de la empresa	INTERNO	1*an

K-Texto = Tabla de ID AI GS1 para ISO/IEC 15961 Formato 9						
92	77	92	92	Información interna de la empresa	INTERNO	1*an
93	78	93	93	Información interna de la empresa	INTERNO	1*an
94	79	94	94	Información interna de la empresa	INTERNO	1*an
95	80	95	95	Información interna de la empresa	INTERNO	1*an
96	81	96	96	Información interna de la empresa	INTERNO	1*an
97	82	97	97	Información interna de la empresa	INTERNO	1*an
98	83	98	98	Información interna de la empresa	INTERNO	1*an
99	84	99	99	Información interna de la empresa	INTERNO	1*an
nnn	85	K-Secundario = S12		AI adicionales		
K-TableEnd = F9B0						

K-Texto = Sec. IDT - Peso neto, kilogramos, libras u onzas troy (artículo comercial de medida variable)						
K-TableID = F9S00						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
310(***)	0	310%x30-36	310%x30-36	Peso neto, kilogramos (artículo comercial de medida variable)	PESO NETO (kg)	6n
320(***)	1	320%x30-36	320%x30-36	Peso neto, libras (artículo comercial de medida variable)	PESO NETO (lb)	6n
356(***)	2	356%x30-36	356%x30-36	Peso neto, onzas troy (artículo comercial de medida variable)	PESO NETO (t)	6n
K-TableEnd = F9S00						

K-Texto = Sec. IDT- Longitud de la primera dimensión (artículo comercial de medida variable)						
K-TableID = F9S01						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
311(***)	0	311%x30-36	311%x30-36	Longitud de la primera dimensión, metros (artículo comercial de medida variable)	LARGO [m]	6n
321(***)	1	321%x30-36	321%x30-36	Longitud de la primera dimensión, pulgadas (artículo comercial de medida variable)	LONGITUD (i)	6n

K-Texto = Sec. IDT- Longitud de la primera dimensión (artículo comercial de medida variable)						
322(***)	2	322%x30-36	322%x30-36	Longitud de la primera dimensión, pies (artículo comercial de medida variable)	LONGITUD (f)	6n
323(***)	3	323%x30-36	323%x30-36	Longitud de la primera dimensión, yardas (artículo comercial de medida variable)	LONGITUD (y)	6n
K-TableEnd = F9S01						

K-Texto = Sec. IDT - Ancho, diámetro o segunda dimensión ,(artículo comercial de medida variable)						
K-TableID = F9S02						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
312(***)	0	312%x30-36	312%x30-36	Ancho, diámetro, o segunda dimensión, metros (artículo comercial de medida variable)	ANCHO (m)	6n
324(***)	1	324%x30-36	324%x30-36	Ancho, diámetro o segunda dimensión, pulgadas (artículo comercial de medida variable)	ANCHO (i)	6n
325(***)	2	325%x30-36	325%x30-36	Ancho, diámetro o segunda dimensión (artículo comercial de medida variable)	ANCHO (f)	6n
326(***)	3	326%x30-36	326%x30-36	Ancho, diámetro o segunda dimensión, yardas (artículo comercial de medida variable)	ANCHO (y)	6n
K-TableEnd = F9S02						

K-Texto = Sec. IDT - Profundidad, grosor, altura o tercera dimensión (artículo comercial de medida variable)						
K-TableID = F9S03						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
313(***)	0	313%x30-36	313%x30-36	Profundidad, grosor, altura o tercera dimensión, metros (artículo comercial de medida variable)	ALTURA (m)	6n
327(***)	1	327%x30-36	327%x30-36	Profundidad, grosor, altura o tercera dimensión, pulgadas (artículo comercial de medida variable)	ALTURA (i)	6n
328(***)	2	328%x30-36	328%x30-36	Profundidad, grosor, altura o tercera dimensión, pies (artículo comercial de medida variable)	ALTURA (f)	6n

K-Texto = Sec. IDT - Profundidad, grosor, altura o tercera dimensión (artículo comercial de medida variable)						
329(***)	3	329%x30-36	329%x30-36	Profundidad, grosor, altura o tercera dimensión, yardas (artículo comercial de medida variable)	ALTURA (y)	6n
K-TableEnd = F9S03						

K-Texto = Sec. IDT - Área (artículo comercial de medida variable)						
K-TableID = F9S04						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
314(***)	0	314%x30-36	314%x30-36	Área, metros cuadrados (artículo comercial de medida variable)	ÁREA (m2)	6n
350(***)	1	350%x30-36	350%x30-36	Área, pulgadas cuadradas (artículo comercial de medida variable)	ÁREA (i2)	6n
351(***)	2	351%x30-36	351%x30-36	Área, pies cuadrados (artículo comercial de medida variable)	ÁREA (f2)	6n
352(***)	3	352%x30-36	352%x30-36	Área, yardas cuadradas (artículo comercial de medida variable)	ÁREA (y2)	6n
K-TableEnd = F9S04						

K-Texto = Sec. IDT - Volumen neto (artículo comercial de medida variable)						
K-TableID = F9S05						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 8						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
315(***)	0	315%x30-36	315%x30-36	Volumen neto, litros (artículo comercial de medida variable)	VOLUMEN NETO (l)	6n
316(***)	1	316%x30-36	316%x30-36	Volumen neto, metros cúbicos (artículo comercial de medida variable)	VOLUMEN NETO (m3)	6n
357(***)	2	357%x30-36	357%x30-36	Peso neto (o volumen), onzas (artículo comercial de medida variable)	VOLUMEN NETO (oz)	6n
360(***)	3	360%x30-36	360%x30-36	Volumen neto, cuartos de galón (artículo comercial de medida variable)	VOLUMEN NETO (q)	6n
361(***)	4	361%x30-36	361%x30-36	Volumen neto, galones estadounidenses (artículo comercial de medida variable)	VOLUMEN NETO (g)	6n
364(***)	5	364%x30-36	364%x30-36	Volumen neto, pulgadas cúbicas	VOLUMEN (i3), registro	6n
365(***)	6	365%x30-36	365%x30-36	Volumen neto, pies cúbicos (artículo comercial de medida variable)	VOLUMEN (f3), registro	6n

K-Texto = Sec. IDT - Volumen neto (artículo comercial de medida variable)						
366(***)	7	366%x30-36	366%x30-36	Volumen neto, yardas cúbicas (artículo comercial de medida variable)	VOLUMEN (y3), registro	6n
K-TableEnd = F9S05						

K-Texto = Sec. IDT - Volumen logístico						
K-TableID = F9S06						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 8						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
335(***)	0	335%x30-36	335%x30-36	Volumen logístico, litros	VOLUMEN (l), reg.	6n
336(***)	1	336%x30-36	336%x30-36	Volumen logístico, metros cúbicos	VOLUMEN (m3), registro	6n
362(***)	2	362%x30-36	362%x30-36	Volumen logístico, cuartos de galón	VOLUMEN (q), reg.	6n
363(***)	3	363%x30-36	363%x30-36	Volumen logístico, galones	VOLUMEN (g), reg.	6n
367(***)	4	367%x30-36	367%x30-36	Volumen logístico, pulgadas cúbicas	VOLUMEN (q), reg.	6n
368(***)	5	368%x30-36	368%x30-36	Volumen logístico, pies cúbicos	VOLUMEN (g), reg.	6n
369(***)	6	369%x30-36	369%x30-36	Volumen logístico, yardas cúbicas	VOLUMEN (i3), registro	6n
K-TableEnd = F9S06						

K-Texto = Sec. IDT - Área de logística						
K-TableID = F9S07						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
334(***)	0	334%x30-36	334%x30-36	Área, metros cuadrados	ÁREA (m2), registro	6n
353(***)	1	353%x30-36	353%x30-36	Área, pulgadas cuadradas	ÁREA (i2), registro	6n
354(***)	2	354%x30-36	354%x30-36	Área, pies cuadrados	ÁREA (f2), registro	6n
355(***)	3	355%x30-36	355%x30-36	Área, yardas cuadradas	ÁREA (y2), registro	6n
K-TableEnd = F9S07						

K-Texto = Sec. IDT - Códigos de cupones						
K-TableID = F9S08						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 8						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString

K-Texto = Sec. IDT - Códigos de cupones						
8100	0	8100	8100	Código extendido del cupón GS1-128 - NSC + Código de oferta	-	6n
8101	1	8101	8101	Código extendido del cupón GS1-128 - NSC + Código de oferta + caducidad del código de oferta		10n
8102	2	8102	8102	Código extendido del cupón GS1-128 - NSC **** OBSOLETO desde GS15i2 ****		2n
8110	3	8110	8110	Código de cupón Identificación para utilizarse en Norteamérica		1*70an
8111	4	8111	8111	Puntos de lealtad de un cupón	PUNTOS	4n
K-TableEnd = F9S08						

K-Texto = Sec. IDT - Longitud o primera dimensión						
K-TableID = F9S09						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
331(***)	0	331%x30-36	331%x30-36	Longitud o primera dimensión, metros	LARGO (m), reg.	6n
341(***)	1	341%x30-36	341%x30-36	Longitud o primera dimensión, pulgadas	LARGO (I), reg.	6n
342(***)	2	342%x30-36	342%x30-36	Longitud o primera dimensión, pies	LARGO (f), reg.	6n
343(***)	3	343%x30-36	343%x30-36	Longitud o primera dimensión, yardas	LARGO (y), reg.	6n
K-TableEnd = F9S09						

K-Texto = Sec. IDT - Ancho, diámetro o segunda dimensión						
K-TableID = F9S10						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
332(***)	0	332%x30-36	332%x30-36	Ancho, diámetro o segunda dimensión, metros	ANCHO (m), reg.	6n
344(***)	1	344%x30-36	344%x30-36	Ancho, diámetro o segunda dimensión	ANCHO (i), reg.	6n
345(***)	2	345%x30-36	345%x30-36	Ancho, diámetro o segunda dimensión	ANCHO (f), reg.	6n
346(***)	3	346%x30-36	346%x30-36	Ancho, diámetro o segunda dimensión	ANCHO (y), reg.	6n
K-TableEnd = F9S10						

K-Texto = Sec. IDT - Profundidad, grosor, altura o tercera dimensión						
K-TableID = F9S11						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 4						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
333(***)	0	333%x30-36	333%x30-36	Profundidad, grosor, altura o tercera dimensión, metros	ALTO (M), reg.	6n
347(***)	1	347%x30-36	347%x30-36	Profundidad, grosor, altura o tercera dimensión	ALTO (i), reg.	6n
348(***)	2	348%x30-36	348%x30-36	Profundidad, grosor, altura o tercera dimensión	ALTO (f), reg.	6n
349(***)	3	349%x30-36	349%x30-36	Profundidad, grosor, altura o tercera dimensión	ALTURA (y), reg.	6n
K-TableEnd = F9S11						

K-Texto = Sec. IDT - AI adicionales						
K-TableID = F9S12						
K-RootOID = urn:oid:1.0.15961.9						
K-IDsize = 128						
AI o AIs	IDvalue	OIDs	IDstring	Nombre	Título de datos	FormatString
243	0	243	243	Número de componente de empaque	PCN	1*20an
255	1	255	255	Número global de cupón	GCN	13*25n
427	2	427	427	Subdivisión del país del código de origen para un artículo comercial	SUBDIVISIÓN DE ORIGEN	1*3an
710	3	710	710	Número de reembolso del sector de salud nacional - Alemania (PZN)	NHRN PZN	3n 1*27an
711	4	711	711	Número de reembolso del sector de salud nacional - Francia (CIP)	NHRN CIP	3n 1*27an

K-Texto = Sec. IDT - AI adicionales						
712	5	712	712	Número de reembolso del sector de salud nacional – España (CN)	NHRN CN	3n 1*27an
713	6	713	713	Número de reembolso del sector de salud nacional – Brasil (DRN)	NHRN DRN	3n 1*27an
8010	7	8010	8010	Identificador de Componente/Pieza	CPID	1*30an

K-Texto = Sec. IDT - AI adicionales						
8011	8	8011	8011	Componente/Pieza Número de serie del identificador	CPID serie	1*12n
8017	9	8017	8017	Número global de relación del servicio - Proveedor	GSRN - PROVEEDOR	18n
8019	10	8019	8019	Número de instancia de relación de servicios	SRIN	1*10n
8200	11	8200	8200	URL extendida de empaque	URL PRODUCTO	1*70an
16	12	16	16	Fecha límite de venta (AAMMDD)	FECHA LÍMITE DE VENTA	6n
394n	13	394%x30-39	394%x30-39	Porcentaje de descuento de un cupón	SIN PCT	4n
7005	14	7005	7005	Zona de captura	ZONA DE CAPTURA	1*12an
7006	15	7006	7006	Fecha de primera congelación	FECHA DE PRIMERA CONGELACIÓN	6n
7007	16	7007	7007	Fecha de cultivo	FECHA DE CULTIVO	6*12an
7008	17	7008	7008	Especies con fines de pesca	ESPECIES ACUÁTICAS	1*3an
7009	18	7009	7009	Tipo de equipo de pesca	TIPO DE EQUIPO DE PESCA	1*10an
7010	19	7010	7010	Método de producción	MÉTODO DE PRODUCCIÓN	1*2an
8012	20	8012	8012	Versión de software	VERSIÓN	1*20an
416	21	416	416	GLN del lugar de producción o de servicio	PROD/SERV/LOC	13n
7020	22	7020	7020	ID de lote de renovación	LOTE RENOV	1*20an
7021	23	7021	7021	Estado funcional	ESTADO FUNC	1*20an
7022	24	7022	7022	Estado de revisión	ESTADO REV	1*20an
7023	25	7023	7023	Identificador Global Individual de Activo (GIAI) de un montaje	GIAI - MONTAJE	1*30an

K-Texto = Sec. IDT - AI adicionales						
235	26	235	235	Extensión serializada de GTIN controlada por terceros	TPX	1*28an
417	27	417	417	Número Global de Localización de la parte	PARTE	13n
714	28	714	714	Número nacional de reembolso de atención sanitaria - Portugal (AIM)	NHRN AIM	1*an20
7040	29	7040	7040	Código de identificación único con extensiones (según EU 2018/574)	UIC	1n 1*3an
8013	30	8013	8013	Número global de modelo	GMN	1*an30
8026	31	8026	8026	Identificación de las piezas de un artículo comercial (ITIP) contenidas en una unidad logística	ITIP CONTENIDO	18n
8112	32	8112	8112	Identificación del código del cupón electrónico para su uso en Norteamérica		1*an70
K-TableEnd = F9S12						

F.2 Formato de valores separados con una coma (CSV)

Esta sección es la tabla de ID de los objetos empaquetados para el formato de datos 9 (Identificadores de aplicación GS1) en formato de valores separados por comas, legibles para máquina, tal como se registra con la ISO. La sección *F.1* es una lista no normativa del contenido de la tabla de los ID para el formato de datos 9, en un formato tabular, legible para el ser humano.

En el formato de valores separados por comas, los saltos de línea tienen significado. Sin embargo, ciertas líneas son demasiado largas para caber dentro de los márgenes de este documento. En la lista que se encuentra a continuación, el símbolo ■ al final de la línea indica que la línea de la tabla de ID continúa en la línea siguiente. Dicha línea se debe interpretar mediante la concatenación de la línea siguiente y la omisión del símbolo ■.

```
K-Text = GS1 AI ID Table for ISO/IEC 15961 Format 9,,,,,,
K-Version = 1.00,,,,,,
K-ISO15434=05,,,,,,
K-Text = Primary Base Table,,,,,,
K-TableID = F9B0,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 90,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
0,1,0,0,SSCC (Serial Shipping Container Code),SSCC,18n
1,2,1,1,Global Trade Item Number,GTIN,14n
02 + 37,3,(2)(37),(02)(37),GTIN + Count of trade items contained in a logistic■
unit,CONTENT + COUNT,(14n)(1*8n)
10,4,10,10,Batch or lot number,BATCH/LOT,1*20an
11,5,11,11,Production date (YYMMDD),PROD DATE,6n
12,6,12,12,Due date (YYMMDD),DUE DATE,6n
13,7,13,13,Packaging date (YYMMDD),PACK DATE,6n
15,8,15,15,Best before date (YYMMDD),BEST BEFORE OR SELL BY,6n
17,9,17,17,Expiration date (YYMMDD),USE BY OR EXPIRY,6n
20,10,20,20,Internal product variant,VARIANT,2n
21,11,21,21,Serial number,SERIAL,1*20an
22,12,22,22,Consumer product variant,CPV,1*20an
240,13,240,240,Additional product identification assigned by the
manufacturer,ADDITIONAL ID,1*30an
241,14,241,241,Customer part number,CUST. PART NO.,1*30an
242,15,242,242,Made-to-Order Variation Number,VARIATION NUMBER,1*6n
250,16,250,250,Secondary serial number,SECONDARY SERIAL,1*30an
```

```
251,17,251,251,Reference to source entity,REF. TO SOURCE ,1*30an
253,18,253,253,Global Document Type Identifier,DOC. ID,13n 0*17an
30,19,30,30,Variable count,VAR. COUNT,1*8n
310n 320n etc,20,K-Secondary = S00,,"Net weight, kilograms or pounds or troy oz
(Variable Measure Trade Item)",,
311n 321n etc,21,K-Secondary = S01,,Length of first dimension (Variable Measure
Trade Item),,
312n 324n etc,22,K-Secondary = S02,,"Width, diameter, or second dimension (Variable
Measure Trade Item)",,
313n 327n etc,23,K-Secondary = S03,,"Depth, thickness, height, or third dimension
(Variable Measure Trade Item)",,
314n 350n etc,24,K-Secondary = S04,,Area (Variable Measure Trade Item),,
315n 316n etc,25,K-Secondary = S05,,Net volume (Variable Measure Trade Item),,
330n or 340n,26,330%x30-36 / 340%x30-36,330%x30-36 / 340%x30-36,"Logistic weight,
kilograms or pounds",GROSS WEIGHT (kg) or (lb),6n / 6n
"331n, 341n, etc",27,K-Secondary = S09,,Length or first dimension,,
"332n, 344n, etc",28,K-Secondary = S10,,"Width, diameter, or second dimension",,
"333n, 347n, etc",29,K-Secondary = S11,,"Depth, thickness, height, or third
dimension",,
334n 353n etc,30,K-Secondary = S07,,Logistic Area,,
335n 336n etc,31,K-Secondary = S06,335%x30-36,Logistic volume,,
337(***),32,337%x30-36,337%x30-36,Kilograms per square metre,KG PER m²,6n
390n or 391n,33,390%x30-39 / 391%x30-39,390%x30-39 / 391%x30-39,Amount payable -
single monetary area or with ISO currency code,AMOUNT,1*15n / 4*18n
392n or 393n,34,392%x30-39 / 393%x30-39,392%x30-39 / 393%x30-39,Amount payable for
Variable Measure Trade Item - single monetary unit or ISO cc, PRICE,1*15n / 4*18n
400,35,400,400,Customer's purchase order number,ORDER NUMBER,1*30an
401,36,401,401,Global Identification Number for Consignment,GINC,1*30an
402,37,402,402,Global Shipment Identification Number,GSIN,17n
403,38,403,403,Routing code,ROUTE,1*30an
410,39,410,410,Ship to - deliver to Global Location Number ,SHIP TO LOC,13n
411,40,411,411,Bill to - invoice to Global Location Number,BILL TO ,13n
412,41,412,412,Purchased from Global Location Number,PURCHASE FROM,13n
413,42,413,413,Ship for - deliver for - forward to Global Location Number,SHIP FOR
LOC,13n
414 and 254,43,(414) [254],(414) [254],"Identification of a physical location GLN,
and optional Extension",LOC No + GLN EXTENSION,(13n) [1*20an]
415 and 8020,44,(415) (8020),(415) (8020),Global Location Number of the Invoicing
Party and Payment Slip Reference Number,PAY + REF No,(13n) (1*25an)
420 or 421,45,(420/421),(420/421),Ship to - deliver to postal code,SHIP TO
POST,(1*20an / 3n 1*9an)
422,46,422,422,Country of origin of a trade item,ORIGIN,3n
423,47,423,423,Country of initial processing,COUNTRY - INITIAL PROCESS.,3*15n
424,48,424,424,Country of processing,COUNTRY - PROCESS.,3n
425,49,425,425,Country of disassembly,COUNTRY - DISASSEMBLY,3n
426,50,426,426,Country covering full process chain,COUNTRY - FULL PROCESS,3n
7001,51,7001,7001,NATO stock number,NSN,13n
7002,52,7002,7002,UN/ECE meat carcasses and cuts classification,MEAT CUT,1*30an
7003,53,7003,7003,Expiration Date and Time,EXPIRY DATE/TIME,10n
7004,54,7004,7004,Active Potency,ACTIVE POTENCY,1*4n
703s,55,7030,7030,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,56,7031,7031,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,57,7032,7032,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,58,7033,7033,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,59,7034,7034,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,60,7035,7035,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,61,7036,7036,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,62,7037,7037,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
```

```
703s,63,7038,7038,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
703s,64,7039,7039,Approval number of processor with ISO country code,PROCESSOR #
s,3n 1*27an
8001,65,8001,8001,"Roll products - width, length, core diameter, direction, and
splices",DIMENSIONS,14n
8002,66,8002,8002,Electronic serial identifier for cellular mobile telephones,CMT
No,1*20an
8003,67,8003,8003,Global Returnable Asset Identifier,GRAI,14n 0*16an
8004,68,8004,8004,Global Individual Asset Identifier,GIAI,1*30an
8005,69,8005,8005,Price per unit of measure,PRICE PER UNIT,6n
8006,70,8006,8006,Identification of the component of a trade item,GCTIN,18n
8007,71,8007,8007,International Bank Account Number ,IBAN,1*30an
8008,72,8008,8008,Date and time of production,PROD TIME,8*12n
8018,73,8018,8018,Global Service Relation Number - Recipient,GSRN - RECIPIENT,18n
8100 8101 etc,74,K-Secondary = S08,,Coupon Codes,,
90,75,90,90,Information mutually agreed between trading partners (including FACT
DIs),INTERNAL,1*30an
91,76,91,91,Company internal information,INTERNAL,1*an
92,77,92,92,Company internal information,INTERNAL,1*an
93,78,93,93,Company internal information,INTERNAL,1*an
94,79,94,94,Company internal information,INTERNAL,1*an
95,80,95,95,Company internal information,INTERNAL,1*an
96,81,96,96,Company internal information,INTERNAL,1*an
97,82,97,97,Company internal information,INTERNAL,1*an
98,83,98,98,Company internal information,INTERNAL,1*an
99,84,99,99,Company internal information,INTERNAL,1*an
nnn,85,K-Secondary = S12,,Additional AIs,,
K-TableEnd = F9B0,,,,,,

"K-Text = Sec. IDT - Net weight, kilograms or pounds or troy oz (Variable Measure
Trade Item)",,,,,,
K-TableID = F9S00,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
310(***),0,310%x30-36,310%x30-36,"Net weight, kilograms (Variable Measure Trade
Item)",NET WEIGHT (kg),6n
320(***),1,320%x30-36,320%x30-36,"Net weight, pounds (Variable Measure Trade
Item)",NET WEIGHT (lb),6n
356(***),2,356%x30-36,356%x30-36,"Net weight, troy ounces (Variable Measure Trade
Item)",NET WEIGHT (t),6n
K-TableEnd = F9S00,,,,,,

K-Text = Sec. IDT - Length of first dimension (Variable Measure Trade Item),,,,,,
K-TableID = F9S01,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
311(***),0,311%x30-36,311%x30-36,"Length of first dimension, metres (Variable
Measure Trade Item)",LENGTH (m),6n
321(***),1,321%x30-36,321%x30-36,"Length or first dimension, inches (Variable
Measure Trade Item)",LENGTH (i),6n
322(***),2,322%x30-36,322%x30-36,"Length or first dimension, feet (Variable Measure
Trade Item)",LENGTH (f),6n
323(***),3,323%x30-36,323%x30-36,"Length or first dimension, yards (Variable
Measure Trade Item)",LENGTH (y),6n
K-TableEnd = F9S01,,,,,,

"K-Text = Sec. IDT - Width, diameter, or second dimension (Variable Measure Trade
Item)",,,,,,
K-TableID = F9S02,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
```

```
312(***),0,312%x30-36,312%x30-36,"Width, diameter, or second dimension, metres
(Variable Measure Trade Item)",WIDTH (m),6n
324(***),1,324%x30-36,324%x30-36,"Width, diameter, or second dimension, inches
(Variable Measure Trade Item)",WIDTH (i),6n
325(***),2,325%x30-36,325%x30-36,"Width, diameter, or second dimension, (Variable
Measure Trade Item)",WIDTH (f),6n
326(***),3,326%x30-36,326%x30-36,"Width, diameter, or second dimension, yards
(Variable Measure Trade Item)",WIDTH (y),6n
K-TableEnd = F9S02,,,,,,

"K-Text = Sec. IDT - Depth, thickness, height, or third dimension (Variable Measure
Trade Item)",,,,,,
K-TableID = F9S03,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
313(***),0,313%x30-36,313%x30-36,"Depth, thickness, height, or third dimension,
metres (Variable Measure Trade Item)",HEIGHT (m),6n
327(***),1,327%x30-36,327%x30-36,"Depth, thickness, height, or third dimension,
inches (Variable Measure Trade Item)",HEIGHT (i),6n
328(***),2,328%x30-36,328%x30-36,"Depth, thickness, height, or third dimension,
feet (Variable Measure Trade Item)",HEIGHT (f),6n
329(***),3,329%x30-36,329%x30-36,"Depth, thickness, height, or third dimension,
yards (Variable Measure Trade Item)",HEIGHT (y),6n
K-TableEnd = F9S03,,,,,,

K-Text = Sec. IDT - Area (Variable Measure Trade Item),,,,,,
K-TableID = F9S04,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
314(***),0,314%x30-36,314%x30-36,"Area, square metres (Variable Measure Trade
Item)",AREA (m2),6n
350(***),1,350%x30-36,350%x30-36,"Area, square inches (Variable Measure Trade
Item)",AREA (i2),6n
351(***),2,351%x30-36,351%x30-36,"Area, square feet (Variable Measure Trade
Item)",AREA (f2),6n
352(***),3,352%x30-36,352%x30-36,"Area, square yards (Variable Measure Trade
Item)",AREA (y2),6n
K-TableEnd = F9S04,,,,,,

K-Text = Sec. IDT - Net volume (Variable Measure Trade Item),,,,,,
K-TableID = F9S05,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 8,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
315(***),0,315%x30-36,315%x30-36,"Net volume, litres (Variable Measure Trade
Item)",NET VOLUME (l),6n
316(***),1,316%x30-36,316%x30-36,"Net volume, cubic metres (Variable Measure Trade
Item)",NET VOLUME (m3),6n
357(***),2,357%x30-36,357%x30-36,"Net weight (or volume), ounces (Variable Measure
Trade Item)",NET VOLUME (oz),6n
360(***),3,360%x30-36,360%x30-36,"Net volume, quarts (Variable Measure Trade
Item)",NET VOLUME (q),6n
361(***),4,361%x30-36,361%x30-36,"Net volume, gallons U.S. (Variable Measure Trade
Item)",NET VOLUME (g),6n
364(***),5,364%x30-36,364%x30-36,"Net volume, cubic inches","VOLUME (i3), log",6n
365(***),6,365%x30-36,365%x30-36,"Net volume, cubic feet (Variable Measure Trade
Item)","VOLUME (f3), log",6n
366(***),7,366%x30-36,366%x30-36,"Net volume, cubic yards (Variable Measure Trade
Item)","VOLUME (y3), log",6n
K-TableEnd = F9S05,,,,,,

K-Text = Sec. IDT - Logistic Volume,,,,,,
K-TableID = F9S06,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
```

```
K-IDsize = 8,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
335(***),0,335%x30-36,335%x30-36,"Logistic volume, litres","VOLUME (l), log",6n
336(***),1,336%x30-36,336%x30-36,"Logistic volume, cubic meters","VOLUME (m3), log",6n
362(***),2,362%x30-36,362%x30-36,"Logistic volume, quarts","VOLUME (q), log",6n
363(***),3,363%x30-36,363%x30-36,"Logistic volume, gallons","VOLUME (g), log",6n
367(***),4,367%x30-36,367%x30-36,"Logistic volume, cubic inches","VOLUME (q), log",6n
368(***),5,368%x30-36,368%x30-36,"Logistic volume, cubic feet","VOLUME (g), log",6n
369(***),6,369%x30-36,369%x30-36,"Logistic volume, cubic yards","VOLUME (i3), log",6n
K-TableEnd = F9S06,,,,,,

K-Text = Sec. IDT - Logistic Area,,,,,,
K-TableID = F9S07,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
334(***),0,334%x30-36,334%x30-36,"Area, square metres","AREA (m2), log",6n
353(***),1,353%x30-36,353%x30-36,"Area, square inches","AREA (i2), log",6n
354(***),2,354%x30-36,354%x30-36,"Area, square feet","AREA (f2), log",6n
355(***),3,355%x30-36,355%x30-36,"Area, square yards","AREA (y2), log",6n
K-TableEnd = F9S07,,,,,,

K-Text = Sec. IDT - Coupon Codes,,,,,,
K-TableID = F9S08,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 8,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
8100,0,8100,8100,GS1-128 Coupon Extended Code - NSC + Offer Code,-,6n
8101,1,8101,8101,GS1-128 Coupon Extended Code - NSC + Offer Code + end of offer code,-,10n
8102,2,8102,8102,GS1-128 Coupon Extended Code - NSC ** DEPRECATED as of GS1GS15i2 **,-,2n
8110,3,8110,8110,Coupon Code Identification for Use in North America,,1*70an
8111,22,8111,8111,Loyalty points of a coupon,POINTS,4n
K-TableEnd = F9S08,,,,,,

K-Text = Sec. IDT - Length or first dimension,,,,,,
K-TableID = F9S09,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
331(***),0,331%x30-36,331%x30-36,"Length or first dimension, metres","LENGTH (m), log",6n
341(***),1,341%x30-36,341%x30-36,"Length or first dimension, inches","LENGTH (i), log",6n
342(***),2,342%x30-36,342%x30-36,"Length or first dimension, feet","LENGTH (f), log",6n
343(***),3,343%x30-36,343%x30-36,"Length or first dimension, yards","LENGTH (y), log",6n
K-TableEnd = F9S09,,,,,,

"K-Text = Sec. IDT - Width, diameter, or second dimension",,,,,,
K-TableID = F9S10,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
332(***),0,332%x30-36,332%x30-36,"Width, diameter, or second dimension, metres","WIDTH (m), log",6n
344(***),1,344%x30-36,344%x30-36,"Width, diameter, or second dimension","WIDTH (i), log",6n
345(***),2,345%x30-36,345%x30-36,"Width, diameter, or second dimension","WIDTH (f), log",6n
```

```
346(***),3,346%x30-36,346%x30-36,"Width, diameter, or second dimension","WIDTH
(y), log",6n
K-TableEnd = F9S10,,,,,,

"K-Text = Sec. IDT - Depth, thickness, height, or third dimension",,,,,,
K-TableID = F9S11,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 4,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
333(***),0,333%x30-36,333%x30-36,"Depth, thickness, height, or third dimension,
metres","HEIGHT (m), log",6n
347(***),1,347%x30-36,347%x30-36,"Depth, thickness, height, or third
dimension","HEIGHT (i), log",6n
348(***),2,348%x30-36,348%x30-36,"Depth, thickness, height, or third
dimension","HEIGHT (f), log",6n
349(***),3,349%x30-36,349%x30-36,"Depth, thickness, height, or third
dimension","HEIGHT (y), log",6n
K-TableEnd = F9S11,,,,,,

K-Text = Sec. IDT - Additional AIs,,,,,,
K-TableID = F9S12,,,,,,
K-RootOID = urn:oid:1.0.15961.9,,,,,,
K-IDsize = 128,,,,,,
AI or AIs,IDvalue,OIDs,IDstring,Name,Data Title,FormatString
243,0,243,243,Packaging Component Number,PCN,1*20an
255,1,255,255,Global Coupon Number,GCN,13*25n
427,2,427,427,Country Subdivision of Origin Code for a Trade Item,ORIGIN
SUBDIVISION,1*3an
710,3,710,710,National Healthcare Reimbursement Number - Germany (PZN),NHRN PZN,3n
1*27an
711,4,711,711,National Healthcare Reimbursement Number - France (CIP),NHRN CIP,3n
1*27an
712,5,712,712,National Healthcare Reimbursement Number - Spain (CN),NHRN CN,3n
1*27an
713,6,713,713,National Healthcare Reimbursement Number - Brazil (DRN),NHRN DRN,3n
1*27an
8010,7,8010,8010,Component / Part Identifier,CPID,1*30an
8011,8,8011,8011,Component / Part Identifier Serial Number,CPID Serial,1*12n
8017,9,8017,8017,Global Service Relation Number - Provider,GSRN - PROVIDER,18n
8019,10,8019,8019,Service Relation Instance Number,SRIN,1*10n
8200,11,8200,8200,Extended Packaging URL,PRODUCT URL,1*70an
16,12,16,16,Sell by date (YYMMDD),SELL BY,6n
394n,13,394%x30-39,394%x30-39,Percentage discount of a coupon,PCT OFF,4n
7005,14,7005,7005,Catch area,CATCH AREA,1*12an
7006,15,7006,7006,First freeze date,FIRST FREEZE DATE,6n
7007,16,7007,7007,Harvest date,HARVEST DATE,6*12an
7008,17,7008,7008,Species for fishery purposes,ACQUATIC SPECIES,1*3an
7009,18,7009,7009,Fishing gear type,FISHING GEAR TYPE,1*10an
7010,19,7010,7010,Production method,PROD METHOD,1*2an
8012,20,8012,8012,Software version,VERSION,1*20an
416,21,416,416,GLN of the production or servie location,PROD/SERV/LOC,13n
7020,22,7020,7020,Refurbishment lot ID,REFURB LOT,1*20an
7021,23,7021,7021,Functional status,FUNC STAT,1*20an
7022,24,7022,7022,Revision status,REV STAT,1*20an
7023,25,7023,7023,Global Individual Assset Identifier (GIAI) of an Assembly,GIAI-
ASSEMBLY,1*30an
235,26,235,235,Third party controlled, serialised extension of GTIN,TPX,1*28n
417,27,417,417,Global Location Number of Party,PGLN,13n
714,28,714,714,National Healthcare Reimbursement Number - Portugal (AIM),NHRH
AIM,1*an20
7040,29,7040,7040,Unique Identification Code with Extensions (per EU 2018/574),UIC,
1n 1*3an
8013,30,8013,8013,Global Model Number,GMN,1*an30
8026,31,8026,8026,Identification of pieces of a trade item (ITIP) contained in a
logistics unit,ITIP CONTENT,18n
8112,32,8112,8112,Paperless coupon code identification for use in North
```

```
America,,1*an70
K-TableEnd = F9S12,,,,,,
```

G Conjunto de caracteres alfanuméricos de 6 bits

La tabla siguiente especifica los caracteres que se usan en la referencia de componente/pieza en los CPI EPC, así como en el número original de la pieza y el número de serie en los ADI EPC. Además, se usa un subconjunto de estos caracteres para el código CAGE/DoAAC en ADI EPC. Las columnas son de la manera siguiente:

- **Símbolo gráfico**: La representación impresa del carácter, tal como se usa en los formatos legibles para los humanos.
- **Nombre**: El nombre común para el carácter
- **Valor binario**: Un numeral binario que ofrece el valor binario de 6 bits para el carácter, tal como se utiliza en las codificaciones binarias de EPC. Este valor binario siempre es igual a los seis bits menos significativos del código de la ISO 646 (ASCII) para el carácter.
- **Forma de URI**: La representación del carácter dentro de las formas de URI de identidad pura EPC y de URI de la etiqueta de EPC. Este es un solo carácter cuyos seis bits menos significativos del código ASCII son iguales al valor en la columna "valor binario" o un triple escape que consta de un carácter de porcentaje seguido de dos caracteres que dan el valor hexadecimal al carácter.

Tabla G-1 Caracteres permitidos en campos alfanuméricos de 6 bits

Símbolo gráfico	Nombre	Valor binario	Forma de URI	Símbolo gráfico	Nombre	Valor binario	Forma de URI
#	Signo de número/libra	100011	%23	H	H mayúscula	001000	H
-	Signo de menos/guion	101101	-	I	Capital I	001001	I
/	Barra de avance	101111	%2F	J	J mayúscula	001010	J
0	Dígito cero	110000	0	K	K mayúscula	001011	K
1	Un dígito	110001	1	L	L mayúscula	001100	L
2	Dos dígitos	110010	2	M	M mayúscula	001101	M
3	Tres dígitos	110011	3	N	N mayúscula	001110	N
4	Cuatro dígitos	110100	4	O	O mayúscula	001111	O
5	Cinco dígitos	110101	5	P	P mayúscula	010000	P
6	Seis dígitos	110110	6	Q	Q mayúscula	010001	Q
7	Siete dígitos	110111	7	R	R mayúscula	010010	R
8	Ocho dígitos	111000	8	S	S mayúscula	010011	S
9	Nueve dígitos	111001	9	T	T mayúscula	010100	T
A	A mayúscula	000001	A	U	U mayúscula	010101	U
B	B mayúscula	000010	B	V	V mayúscula	010110	V
C	C mayúscula	000011	C	W	W mayúscula	010111	W
D	D mayúscula	000100	D	X	X mayúscula	011000	X
E	E mayúscula	000101	E	Y	Y mayúscula	011001	Y
F	F mayúscula	000110	F	Z	Letra Z mayúscula	011010	Z
G	G mayúscula	000111	G				

H (Omitido de manera intencional)

[Se omite este apéndice de tal manera que los Apéndices I al M, que especifican los objetos empaquetados, tienen las mismas letras de apéndice correspondientes a los anexos de la ISO/IEC 15962 , 2a Edición].

I Estructura de objetos empaquetados

I.1 Descripción general

El formato de los Objetos Empaquetados proporciona una codificación eficiente y acceso a los datos del usuario. El formato de los Objetos Empaquetados ofrece eficiencia de la codificación incrementada en comparación con los métodos de acceso sin directorio y con directorio, en parte, al utilizar métodos sofisticados de compactación, y en parte, al definir una estructura inherente del directorio, al frente de cada Objeto Empaquetado (antes de que alguno de sus datos se haya codificado), lo que permite el acceso aleatorio, mientras reduce el espacio superior fijo de algunos métodos previos, y finalmente, en parte, al usar la información específica del sistema de datos (como las definiciones de GS1 de los identificadores de aplicación de longitud fija).

I.2 Descripción general de la documentación de Objetos Empaquetados

La descripción formal de los Objetos Empaquetados se presenta en este Apéndice y en los Apéndices J, K, L y M, como se indica a continuación:

- La estructura general de los Objetos Empaquetados se describe en la *Sección I.3*.
- Las secciones individuales de un Objeto Empaquetado se describen en las Secciones *I.4* a la *I.9*.
- La estructura y las características de las Tablas de ID (utilizadas por los Objetos Empaquetados para representar diferentes identificadores del sistema de datos) se describen en el *Apéndice J*.
- Las bases numéricas y los conjuntos de caracteres en Objetos Empaquetados se describen en el *Apéndice K*.
- Un algoritmo de codificación y un ejemplo trabajado se describen en el *Apéndice L*.
- El algoritmo de decodificación para Objetos Empaquetados se describe en el *Apéndice M*.

Además, cabe mencionar que todas las descripciones de las Tablas de ID específicas se registran por separado para su uso con Objetos Empaquetados, según los procedimientos de la ISO/IEC 15961-2, tal como la descripción formal completa del formato legible para máquinas para las Tablas de ID registradas.

I.3 Diseño del formato de Objetos Empaquetados de alto nivel

I.3.1 Descripción general

El formato de la memoria de los Objetos Empaquetados consta de una secuencia en memoria de una o más estructuras de datos de los "Objetos Empaquetados". Cada Objeto Empaquetado puede contener o datos codificados o información del directorio, pero no ambos. El primer Objeto Empaquetado en memoria va precedido por un DSFID. El DSFID indica el uso de Objetos Empaquetados como el Método de Acceso de la memoria, e indica el Formato de datos registrado que es el formato predeterminado para cada Objeto Empaquetado en dicha memoria. Cada Objeto Empaquetado puede ir precedido o seguido de patrones de relleno (si es necesario, para el alineamiento en los límites de palabras o bloques). Además, como máximo un Objeto Empaquetado en memoria puede ir precedido por un puntero a un Objeto Empaquetado con directorio (este puntero puede ir seguido de relleno). Esta serie de Objetos Empaquetados termina con un relleno opcional seguido de uno o más octetos de valor cero alineados en los límites de bytes. Consulte la *Figura 13-1*, que muestra esta secuencia cuando aparece en una etiqueta RFID.

Nota: *Dado que las estructuras de datos de un Objeto Empaquetado codificado están alineadas en bits en lugar de en bytes, en este apéndice se utiliza el término 'octeto' en lugar de 'byte', excepto en caso de que se deba alinear una cantidad de ocho bits en un límite de bytes.*

Figura I-1 Estructura general de la memoria al utilizar Objetos Empaquetados

DSFID	Puntero Opcional* y/o relleno	Primer Objeto Empaquetado	Puntero Opcional* y/o relleno	Segundo Objeto Empaquetado Opcional	…	Objeto Empaquetado Opcional	Puntero Opcional* y/o relleno	Octeto(s) cero

* Nota: el Puntero Opcional a un Objeto Empaquetado de Directorio puede aparecer como máximo solo una vez en memoria

Cada Objeto Empaquetado representa una secuencia de uno o más identificadores del sistema de datos, cada uno especificado por referencia a una entrada dentro de una Tabla de ID Base desde un formato de datos registrado. Se hace referencia a la entrada mediante su posición relativa dentro de la Tabla Base; a esta posición relativa o índice de Tabla Base se hace referencia en toda esta especificación como un "Valor ID". Hay dos métodos diferentes de Objetos Empaquetados disponibles para representar una secuencia de identificadores por referencia a sus valores de ID:

- Un Objeto Empaquetado de lista de ID (IDLPO) codifica una serie de valores de ID como una lista, cuya longitud depende de la cantidad de elementos de datos que se representan;
- En su lugar, un Objeto Empaquetado de Mapas de ID (IDMPO) codifica una matriz de bits de longitud fija, cuya longitud depende de la cantidad total de entradas definidas en la Tabla Base registrada. Cada bit de la matriz es '1' si la entrada de la tabla correspondiente está representada por el Objeto Empaquetado; de lo contrario, es '0'.

Una Lista de ID es el formato predeterminado de Objetos Empaquetados, porque utiliza menos bits que un Mapa de ID, si la lista contiene solo un pequeño porcentaje de los valores de ID definidos del sistema de datos. Sin embargo, si el Objeto Empaquetado incluye más de una cuarta parte de las entradas definidas, un Mapa de ID requiere menos bits. Por ejemplo, si un sistema de datos tiene dieciséis entradas, cada valor de ID (índice de tabla) es una cantidad de cuatro bits y una lista de cuatro valores de ID toma tantos bits como el mapa de ID completo. La característica de longitud fija de un mapa de ID lo hace especialmente adecuado para su uso en un Objeto Empaquetado con Directorio, que enumera todos los identificadores en todos los Objetos Empaquetados en memoria (consulte la Sección n *I.9*). La estructura general de un Objeto Empaquetado es la misma, ya sea un IDLPO o un IDMPO, como se muestra en la Figura I 3-2 y como se describe en la siguiente subsección.

Figura I-2 Estructura de objetos empaquetados

Indicadores de formato opcionales	Sección de información del objeto (**IDLPO** o **IDMPO**)	Sección de ID secundaria (si es necesario)	Sección Formato Aux (si es necesario)	Sección de datos (si es necesario)

Los objetos empaquetados se pueden hacer "editables", agregando una subsección Apéndice opcional al final de la sección de Información del Objeto, que incluye un puntero a un "Objeto Empaquetado de Apéndice" donde se han realizado adiciones y/o eliminaciones. Una o más de esas "cadenas" de Objetos Empaquetados "principales" y "secundarios" editables pueden estar presentes en la secuencia general de Objetos Empaquetados en memoria, pero no puede haber más de una cadena de Objetos Empaquetados con directorio.

I.3.2 Descripción de cada sección de la estructura de un Objeto Empaquetado

Cada Objeto Empaquetado consta de varias secciones alineadas en bits (es decir, no se utilizan bits de relleno entre secciones), transportadas en un número variable de octetos. Todos los formatos obligatorios y opcionales de los Objetos Empaquetados están incluidos en la siguiente lista ordenada de secciones de Objetos Empaquetados. Después de esta lista, se presenta cada sección de Objetos Empaquetados, y en las secciones posteriores de este Anexo se describe detalladamente cada sección de Objetos Empaquetados.

- **Indicadores de formato**: Un Objeto Empaquetado puede comenzar opcionalmente con el patrón '0000' que está reservado para introducir uno o más indicadores de formato, como se describe en *I.4.2*. Estos indicadores pueden indicar el uso del formato del mapa de ID no predeterminado. Si los Indicadores de Formato no están presentes, el Objeto Empaquetado toma de forma predeterminada el formato de la Lista de ID.
  - Ciertos patrones de indicadores indican un patrón entre Objetos (Puntero de Directorio o Relleno)
  - Otros patrones de indicadores indican el tipo de Objetos Empaquetados (Mapa o Lista), y pueden indicar la presencia de una subsección de Apéndice opcional para su edición.
- **Información del objeto:** Todos los Objetos Empaquetados contienen una sección de Información del Objeto que incluye Información de Longitud de Objeto e Información de Valor de ID:
  - La Información de longitud del objeto incluye un campo ObjectLength (que indica la longitud total del Objeto Empaquetado en octetos) seguido por el bit indicador de Pad, de modo que se pueda determinar el número de bits significativos en el Objeto Empaquetado.
  - La información del valor de ID indica qué identificadores están presentes y en qué orden, y (si se trata de un IDLPO) también incluye un campo NumberOfIDs inicial, que indica cuántos valores de ID se codifican en la lista de ID.

La sección Información del Objeto está codificada en uno de los siguientes formatos, como se muestra en la *Figura I-3* y *Figura I-4*.

- Formato de Información del Objeto de la Lista de ID (IDLPO):
  - Longitud del objeto (EBV-6) más bit Indicador de Almohadilla
  - Una única lista de ID o una sección de listas de ID (dependiendo de los Indicadores de Formato)
- Formato de la Información del Objeto del Mapa de ID (IDMPO):
  - Una o más secciones de Mapa de ID
  - Longitud del objeto (EBV-6) más bit Indicador de Almohadilla

Para cualquiera de estos formatos de Información de Objeto, puede haber una subsección de Apéndice opcional al final de la sección de Información del Objeto.

- **Bits de ID secundarios**: Un Objeto Empaquetado puede incluir una sección de ID Secundario, si es necesario, para codificar bits adicionales definidos para algunas clases de ID (estos bits completan la definición del ID.).
- **Bits de Formato Aux:** Un Objeto Empaquetado de Datos puede incluir una sección de Formato Auxiliar, que, si está presente, codifica uno o más bits definidos para admitir la compresión de datos, pero que no contribuye con la definición del ID.
- **Sección de datos:** Un Objeto Empaquetado de Datos incluye una Sección de Datos, que representa los datos comprimidos asociados a cada uno de los identificadores enumerados en el Objeto Empaquetado. Esta sección se omite en un Objeto Empaquetado con Directorio, y en un Objeto Empaquetado que utiliza la compactación sin directorio (consulte *I.7.1*). En función de la declaración del formato de datos de la tabla de ID correspondiente, la sección Datos contendrá una o ambas subsecciones:
  - **Subsección Numéricos de Longitud conocida:** esta subsección compacta y concatena todas las cadenas de datos no vacías que se conocen a priori como numéricas.
  - **Subsección Alfanumérica:** esta subsección concatena y compacta todas las cadenas de datos no vacías que no se conocen a priori como completamente numéricas.

Figura I-3 Estructura de Información del Objeto IDLPO

Información de objeto, en una PO de lista de ID predeterminada					Información de Objeto, en una PO de la lista de ID no predeterminada		
Longitud del Objeto	Número de ID	Lista de ID	Apéndice opcional	o	Longitud del Objeto	Sección de Listas de ID (una o más listas)	Apéndice opcional

Figura I-4 Estructura de Información del Objeto IDMPO

Información del Objeto, en una PO de Mapa de ID		
Sección de Mapa de ID (una o más listas)	Longitud del Objeto	Apéndice opcional

I.4 Sección de indicadores de formato

El diseño predeterminado de la memoria, según el método de acceso de Objetos Empaquetados, consta de un DSFID inicial, seguido inmediatamente de un Objeto Empaquetado de la Lista de ID (en el siguiente límite de bytes) y, a continuación, de manera opcional, Objetos Empaquetados de Lista de ID adicionales (cada uno comenzando en el siguiente límite de bytes), y terminado por un octeto de valor cero en el siguiente límite de bytes (que indica que no se codifica ningún Objeto Empaquetado adicional). En esta sección se definen los patrones de Indicadores de Formato válidos que pueden aparecer al inicio esperado de un Objeto Empaquetado para reemplazar el diseño predeterminado, si se desea (por ejemplo, cambiando el formato del Objeto Empaquetado, o insertando patrones de relleno para alinear el siguiente Objeto Empaquetado en una palabra o límite de bloques). A continuación se muestra el conjunto de patrones definidos.

Tabla I-1 Indicador de formato

Patrón de bits	Descripción	Información adicional	Ver sección
0000 0000	Patrón de terminación	No siguen más Objetos Empaquetados	*I.4.1*
LLLLLL xx	Primer octeto de un IDLPO	Para cualquier LLLLLLLL > 3	*I.5*
0000	Patrón inicial de Indicadores de Formato	(si el EBV-6 completo no es cero)	*I.4.2*
0000 10NA	IDLPO con: N = 1: Información no predeterminada A = 1: Apéndice presente	Si N = 1: permite varias tablas de ID Si A = 1: Ptr (s) de Apéndice al final de la sección Información del Objeto	*I.4.3*

Patrón de bits	Descripción	Información adicional	Ver sección
0000 01xx	Patrón entre OP	Un Puntero con Directorio o relleno	*I.4.4*
0000 0100	Significa un octeto de relleno	A continuación no aparece ningún indicador de longitud de relleno	*I.4.4*
0000 0101	Significa relleno de longitud de ejecución	A continuación se muestra una longitud de relleno EBV-8	*I.4.4*
0000 0110	RFU		*I.4.4*
0000 0111	Puntero con directorio	Seguido del patrón EBV-8	*I.4.4*
0000 11xx	Objeto Empaquetado del Mapa de ID		*I.4.2*
0000 0001 0000 0010 0000 0011	[No válida]	Patrón no válido	

I.4.1 Patrón del indicador final de datos

Un patrón de ocho o más bits '0' al inicio esperado de un Objeto Empaquetado indica que no hay más Objetos Empaquetados presentes en el resto de la memoria.

NOTA: Seis bits '0' sucesivos al inicio esperado de un Objeto Empaquetado (si se interpreta como un Objeto Empaquetado) indicarían un Objeto Empaquetado de la Lista de ID de longitud cero.

I.4.2 Patrones de bits iniciales de la sección de indicadores de formato

Un EBV-6 distinto de cero con un patrón inicial de “0000” se utiliza como un Patrón de Indicación de sección de Indicadores de Formato. Los bits adicionales que siguen a un Patrón Indicador de formato '0000' inicial se definen de la siguiente manera:

- Un patrón siguiente de dos bits de '10' (creando un patrón inicial de '000010') indica un IDLPO con al menos una función opcional no predeterminada (consulte *I.4.3*)
- Un siguiente patrón de dos bits de '11' indica un IDMPO, que es un Objeto Empaquetado que utiliza un formato de Mapa de ID en lugar del Formato de Lista de ID. La sección de Mapa de ID (consulte *I.9*) sigue inmediatamente este patrón de dos bits.
- Un patrón siguiente de dos bits de '01' significa un patrón Externo (patrón de Relleno o Puntero) antes del inicio del siguiente Objeto Empaquetado (consulte *I.4.4*)

Una Longitud de Objeto EBV-6 inicial inferior a cuatro no es válida como una longitud de Objetos Empaquetados.

Nota: El objeto empaquetado más corto posible es un IDLPO, para un sistema de datos que utiliza cuatro bits por valor de ID, codificando un único valor de ID. Este Objeto Empaquetado tiene un total de 14 bits fijos. Por lo tanto, un Objeto Empaquetado de dos octetos solo contendría dos bits de datos y no es válido. Un Objeto Empaquetado de tres octetos podría codificar un único elemento de datos de hasta tres dígitos. Para conservar “3” como una longitud no válida en este escenario, el codificador de Objetos Empaquetados codificará una sección de Indicadores de Formato iniciales (con todas las opciones configuradas en cero, si así se desea) con el fin de aumentar a 4 la longitud del objeto.

I.4.3 Indicadores de Formato IDLPO

La aparición de '000010' al comienzo esperado de un Objeto Empaquetado va seguida de dos bits adicionales, para formar una sección completa de Indicadores de Formato IDLPO de “000010NA”, donde:

- Si el primer bit adicional 'N' es '1', entonces, se utiliza un formato no predeterminado para la sección de Información del Objeto IDLPO. Mientras que el formato IDLPO predeterminado solo permite una única Lista de ID (utilizando la Tabla de ID base predeterminada del registro), el formato de Información del Objeto IDLPO no predeterminado, opcional admite una secuencia de una o más listas de ID, y cada una de estas listas comienza con la información de identificación de la tabla registrada que representa (consulte *I.5.1*).
- Si el segundo bit adicional “A” es “1”, entonces, se presenta una subsección de Apéndice al final de la sección de Información del Objeto (consulte *I.5.6*).

I.4.4 Patrones de uso entre Objetos Empaquetados

La aparición de '000001' al inicio esperado de un Objeto Empaquetado se utiliza para indicar relleno o puntero con directorio, de la siguiente manera:

- Un patrón siguiente de dos bits de '11' indica que el Puntero del Objeto Empaquetado con directorio sigue el patrón. El puntero tiene una o más octetos de longitud, en formato EBV-8. Este puntero puede ser Nulo (un valor de cero), pero si no es cero, indica el número de octetos desde el inicio del puntero hasta el inicio de un Objeto Empaquetado con directorio (que, si se puede editar, será el primero de su "cadena"). Por ejemplo, si el byte de los Indicadores de un Puntero con Directorio está codificado en el desplazamiento de bytes 1, en sí el Puntero ocupa bytes que comienzan en el desplazamiento 2, y el Directorio comienza en el desplazamiento de bytes 9, entonces, el Ptr Dir codifica el valor "7" en formato EBV-8. Un Puntero de Objeto Empaquetado con Directorio puede aparecer antes del primer Objeto Empaquetado en memoria o en cualquier otra posición en la que pueda comenzar un Objeto Empaquetado, pero solo puede aparecer una vez en una memoria portadora de datos determinada. Además (si no es nula), debe encontrarse en una dirección inferior a la del Directorio al que apunta. El primer octeto después de este puntero puede ser relleno (como se define inmediatamente a continuación), un nuevo conjunto de patrones de los Indicadores de Formato o el inicio de un Objeto Empaquetado de la Lista de ID.
- Un patrón siguiente de dos bits de '00' indica que el patrón completo de ocho bits de '00000100' sirve como byte de relleno, de modo que el siguiente Objeto Empaquetado puede comenzar en un límite de canales o bloques deseado. Este patrón puede repetirse según sea necesario para lograr la alineación deseada.
- Un patrón siguiente de dos bits de '01' como un indicador de relleno de longitud de ejecución, y será inmediatamente seguido por un EBV-8 que indique la cantidad de octetos desde el inicio del propio EBV-8 hasta el inicio del siguiente Objeto Empaquetado (por ejemplo, si el siguiente Objeto Empaquetado sigue inmediatamente, el EBV-8 tiene un valor de uno). Este mecanismo elimina la necesidad de escribir muchas palabras de memoria para extraer un bloque de memoria grande.
- Se reserva el siguiente patrón de dos bits de '10'.

I.5 Sección de información del objeto

La sección Información del Objeto de cada Objeto Empaquetado contiene tanto Información sobre la longitud (el tamaño del Objeto Empaquetado, en bits y en octetos) como Información de valores de ID. Un Objeto Empaquetado codifica las representaciones de uno o más Identificadores del sistema de datos y (si se trata de un Objeto Empaquetado de Datos) también codifica sus elementos de datos asociados (cadenas AI, cadenas DI, etc.). La información de valores de ID codifica una lista completa de todos los Identificadores (AIS, DIS, etc.) codificados en el Objeto Empaquetado, o (en un Objeto Empaquetado con Directorio) todos los Identificadores codificados en cualquier lugar de la memoria.

Para conservar los bits codificados y transmitidos, los Identificadores del sistema de datos (cada uno representado normalmente en los sistemas de datos por dos, tres o cuatro caracteres ASCII) se representan dentro de un objeto empaquetado mediante un valor de ID, que representa un índice que indica una entrada en una Tabla Base registrada de Valores de ID. Un único Valor de ID puede representar un único Identificador de Objetos o puede representar una secuencia de Identificadores de Objetos que se utiliza habitualmente. En algunos casos, el valor ID representa una "clase" de Identificadores de Objetos relacionados, o una secuencia de Identificadores de Objetos en la que uno o más Identificadores de Objetos se codifican opcionalmente. En estos casos, los bits de ID secundarios (consulte *I.6*) se codifican para especificar qué selección u opción se eligió cuando se codificó el Objeto Empaquetado. Un "valor de ID completo" (FQIDV) es un valor de ID, más una opción particular de bits de ID secundarios asociados (si los hay, se invoca mediante la entrada de tabla del valor de ID). Únicamente puede aparecer una instancia de un valor de ID completo determinado en los Objetos Empaquetados de Datos de un operador de datos; sin embargo, un valor de ID determinado puede aparecer más de una vez, si cada vez se califica mediante diferentes bits de ID secundarios. Si un Valor de ID aparece más de una vez, todas las apariciones se deben realizar en un solo Objeto Empaquetado (o dentro de una única "cadena" de un Objeto Empaquetado más su Apéndice).

Existen dos métodos definidos para codificar Valores de ID: un Objeto Empaquetado de la lista de ID utiliza una lista de longitud variable de campos de bit de Valor de ID, mientras que un Objeto Empaquetado del Mapa de ID utiliza una matriz de bits de longitud fija. A menos que un patrón de Indicadores de formato inicial modifique un formato de Objetos Empaquetados, el formato predeterminado del Objeto Empaquetado es el de un Objeto Empaquetado de la Lista de ID (IDLPO), que contiene una única lista de ID, cuyos valores de ID corresponden a la Tabla Base de ID predeterminada del formato de datos registrado. Los Indicadores de Formato opcionales pueden cambiar el formato de la sección de ID a un formato IDMPO o a un formato IDLPO que codifica una sección de listas de ID (que admite varias Tablas de ID, incluidos sistemas de datos no predeterminados).

Aunque el orden de la información dentro de la sección Información del Objeto varía según el formato que se elija (consulte *I.5.1*), la sección Información del Objeto de cada Objeto Empaquetado proporcionará información sobre la longitud tal como se define en *I.5.2*, e información de valores de ID (consulte *I.5.3)* tal como se define en *I.5.4* o *I.5.5*.

La sección Información del Objeto (de un IDLPO o de un IDMPO) puede concluir con una subsección de Apéndice opcional (consulte *I.5.6)*.

I.5.1 Formatos de información del objeto

I.5.1.1 Formato predeterminado de Información de Objeto DLPO

El formato de Información de Objeto IDLPO predeterminado se utiliza para un Objeto Empaquetado sin una sección de Indicadores de Formato inicial o con una sección de Indicadores de Formato que indica un IDLPO con un posible Apéndice y una sección de Información de Objeto predeterminada. La sección Información de Objeto IDLPO predeterminada contiene una única Lista de ID (seguida opcionalmente por una subsección de Apéndice si así lo indican los Indicadores de Formato). El formato de la sección Información del Objeto IDLPO predeterminado se muestra en la tabla siguiente.

Tabla I-2 Formato de información de objeto IDLPO predeterminado

Nombre del campo:	Información sobre la longitud	NumberOfIDs	Lista de ID	Subsección del apéndice
Uso:	El número de octetos de este Objeto, más el indicador de panel del último octeto	de valores ID en este Objeto (menos uno)	Una única lista de valores de ID; el tamaño del valor depende del Formato de Datos registrado	Puntero(s) opcional(s) a otros Objetos que contienen información sobre Edición
Estructura:	Variable: consulte *I.5.2*	Variable:EBV-3	Consulte *I.5.4*	Consulte *I.5.6*

En la sección Información de Objeto de IDLPO, el campo NumberOfIDs es un Vector de Bits Extensible EBV-3, que consta de una o más repeticiones de un Bit de Extensión seguido de 2 bits de valor. Este EBV-3 codifica uno menos que el número de Valores de ID en la Lista de ID asociado. Por ejemplo, un EBV-3 de '101 000' indica (4 + 0 +1) = 5 valores de ID. La Información de Longitud se describe en *I.5.2* para todos los Objetos Empaquetados. Los siguientes campos son una Lista de ID (consulte *I.5.4)* y una subsección opcional del apéndice (consulte *I.5.6*).

I.5.1.2 Formato predeterminado de Información de Objeto IDLPO

Los Indicadores de Formato principales pueden modificar la estructura de Información de Objeto de un IDLPO, de modo que pueda contener más de una lista de ID, en una sección de listas de ID (que también permite utilizar tablas de ID no predeterminadas). En la tabla siguiente se muestra la estructura de Información de Objeto IDLPO no predeterminada.

Tabla I-3 Formato de Información de Objeto IDLPO no predeterminado

Nombre del campo:	Información sobre la longitud	Sección Listas de ID, Primera lista			Listas de ID adicionales opcionales	Indicador de aplicación nulo (bit cero único)	Subsección del apéndice
		Indicador de aplicación	Número de ID	Lista de ID			
Uso:	El número de octetos de este Objeto, más el indicador de panel del último octeto	Indica la tabla de ID seleccionada y el tamaño de cada entrada	Cantidad de valores de ID en la lista (menos uno)	Lista de Valores de ID y, a continuación, un bit de uso de F/R.	Cero o más listas repetidas, cada una para una Tabla de ID diferente		Puntero(s) opcional(s) a otros Objetos que contienen información sobre Edición
Estructura:	consulte ***1.5.2***	consulte ***I.5.3.1***	Consulte ***I.5.1.1***	Consulte ***I.5.4*** y ***I.5.3.2***	Referencias en columnas anteriores	Consulte ***I.5.3.1***	Consulte ***I.5.6***

I.5.1.3 Formato de Información de Objeto IDMPO

Los Indicadores de Formato Principales pueden definir la estructura de Información del Objeto como un IDMPO, en el que la Información de Longitud (y la subsección Apéndice opcional) siguen una sección de Mapa de ID (consulte *I.5.5*). Esta disposición garantiza que el Mapa de ID se encuentre en una ubicación fija para una aplicación determinada, lo que resulta beneficioso cuando se utiliza como Directorio. En la tabla siguiente se muestra la estructura de la Información del Objeto IDMPO.

Tabla I-4 Formato de Información de Objeto IDMPO

Nombre del campo:	sección de Mapa de ID	Información sobre la longitud	Apéndice
Uso:	Una o más estructuras de Mapa de ID, cada una de ellas con una Tabla de ID diferente	El número de octetos de este Objeto, más el indicador de panel del último octeto	Puntero(s) opcional(s) a otros Objetos que contienen información sobre Edición
Estructura:	consulte *1.9.1*	Consulte *I.5.2*	Consulte *I.5.6*

1.5.2 Información sobre la longitud

En la tabla siguiente se muestra el formato de la información de longitud, siempre presente en la sección Información del Objeto de cualquier Objeto Empaquetado.

Tabla I-5 Información de Longitud del Objeto Empaquetado

Nombre del campo:	ObjectLength	Indicador de relleno
Uso:	el número de bytes de 8 bits en este objeto incluye el primer byte de este Objeto Empaquetado, incluidos sus indicadores de formato IDLPO/IDMPO, si están presentes. Además, excluye los patrones que se utilizan entre Objetos Empaquetados, como se especifica en *I.4.4*	Si '1': el último byte del Objeto contiene al menos 1 relleno
Estructura:	Variable: EBV-6	Fijo: 1 bit

El primer campo, ObjectLength, es un Vector de Bits Extensible EBV-6, que consta de una o más repeticiones de un Bit de Extensión y 5 bits de valor. Un EBV-6 de '000100' (valor de 4) indica un Objeto Empaquetado de cuatro bytes, un EBV-6 de '100001 000000' (valor de 32) indica un Objeto de 32 bytes, etc.

El bit Indicador de Relleno sigue inmediatamente al final de ObjectLength EBV-6. Este bit se establece en '0' si no hay bits de relleno en el último byte del Objeto Empaquetado. Si se establece en '1', el relleno a nivel de bits comienza con el bit '1' menos significativo o más a la derecha del último byte, y el relleno consiste en este bit '1' más a la derecha, más cualquier bit '0' a la derecha de ese bit. Este método utiliza de forma eficaz un ***solo*** bit para indicar una cantidad de ***tres*** bits (es decir, la cantidad de bits de relleno final). Cuando un sistema receptor desea determinar la cantidad total de bits (en lugar de bytes) en un Objeto Empaquetado, examinaría el campo ObjectLength del Objeto Empaquetado (para determinar la cantidad de bytes) y multiplicaría el resultado por ocho, y (si el bit del Indicador de Relleno está establecido) examine el último byte del Objeto Empaquetado y disminuya el recuento de bits en (1 más el número de bits '0' después del bit '1' más a la derecha de ese byte final).

I.5.3 Descripción general de los valores de ID

Un formato de datos registrado define (como mínimo) una Tabla de ID de base primaria (en el Anexo *J*) se puede encontrar una especificación detallada para los cuadros de ID registrados). Esta tabla base define los Identificadores del sistema de datos representados por cada fila de la tabla, cualquier Bit de ID Secundario o bits de Formato Auxiliar invocados por cada entrada de tabla, y varias reglas implícitas (tomadas de un conjunto de reglas predefinido) que los sistemas de decodificación utilizarán al interpretar los datos codificados según cada entrada. Cuando un elemento de datos se codifica en un Objeto Empaquetado, su entrada de tabla asociada se identifica mediante la posición correspondiente de la entrada en la Tabla Base. Esta posición o índice de tabla es el Valor de ID que se representa en Objetos Empaquetados.

Una Tabla Base que contiene un número determinado de entradas especifica de forma inherente la cantidad de bits necesarios para codificar un índice de tabla (es decir, un valor de ID) en un Objeto Empaquetado de la Lista de ID (como el Registro (base 2) del número de entradas). Dado que las Tablas de ID del sistema de datos actuales y futuras variarán de forma impredecible en cuanto al número de entradas de la tabla, es necesario predefinir un mecanismo de Tamaño del Valor de ID que permita la extensibilidad futura para adaptar nuevas tablas, minimizando la complejidad del decodificador y minimizando la necesidad de actualizar el software de decodificación (aparte de la adición de nuevas tablas). Por lo tanto, independientemente del número exacto de entradas definidas de la Tabla Base, cada definición de la tabla base utilizará uno de los tamaños predefinidos para las codificaciones de valores de ID definidas en la Tabla I 5-5 (las entradas no utilizadas se etiquetarán como reservadas, tal como se indica en el Anexo *J*). El patrón de los Bits de Tamaño de ID se codifica en un Objeto Empaquetado únicamente cuando utiliza una Tabla de ID Base no predeterminada. Algunas entradas en la tabla indican un tamaño que no es una potencia integral de dos.

Al codificar (en un IDLPO) los Valores de ID de las tablas que utilizan dichos tamaños, cada par de Valores de ID se codifica multiplicando el ID anterior del par mediante la base especificada en la cuarta columna de la Tabla I-5-5 y agregando el ID posterior del par, y codificando el resultado en el número de bits especificado en la cuarta columna. Si existe un valor de ID único final para esta Tabla de ID, se codifica en el número de bits especificado en la tercera columna de la tabla siguiente.

Tabla I-6 Tamaños de Valores de ID definidos

Patrón de Bits de Tamaño de ID		Cantidad máxima de Entradas de la Tabla	Cantidad de Bits por Valor de ID único o final y cómo codificarlos	Cantidad de Bits por par de Valores de ID y cómo codificarlos
000		Hasta 16	4, como 1 base 16 valores	8, como 2 base 16 valores
001		Hasta 22	5, como 1 base 22 valores	9, como 2 base 22 valores
010		Hasta 32	5, como 1 base 32 valores	10, como 2 base 32 valores
011		Hasta 45	6, como 1 base 45 valores	11, como 2 base 45 valores
100		Hasta 64	6, como 1 base 64 valores	12, como 2 base 64 valores
101		Hasta 90	7, como 1 base 90 valores	13, como 2 base 90 valores
110		Hasta 128	7, como 1 base 128 valores	14, como 2 base 128 valores
1110		Hasta 256	8, como 1 base 256 valores	16, como 2 base 256 valores
111100		Hasta 512	9, como 1 base 512 valores	18, como 2 base 512 valores
111101		Hasta 1024	10, como 1 base 1024 valores	20, como 2 base 1024 valores
111110		Hasta 2048	11, como 1 base 2048 valores	22, como 2 base 2048 valores
111111		Hasta 4096	12, como 1 base 4096 valores	24, como 2 base 4096 valores

I.5.3.1 Subsección de indicadores de la aplicación

Se puede utilizar una subsección de Indicadores de la Aplicación para indicar el uso de Valores de ID de una Tabla de ID predeterminada o no predeterminada. Esta subsección es necesaria en todos los IDMPO, pero es necesaria únicamente en un IDLPO que utiliza el formato no predeterminado que admite varias Listas de ID.

Un Indicador de la Aplicación consta de los componentes siguientes:

- Un solo bit AppIndicatorPresent, que si es '0' significa que no va seguido de ninguna Lista de ID ni Mapa adicional. Tener en cuenta que este bit siempre se omite para la primera Lista o Mapa de una sección de Información del Objeto. Cuando este bit está presente y es '0', no se codifica ninguno de los siguientes campos de bit.
- Un solo bit ExternalReg que, si es '1', indica el uso de una Tabla ID de un registro distinto del predeterminado en la memoria. Si es '1', este bit va seguido inmediatamente de una representación de 9 bits de un Formato de Datos registrado en la ISO/IEC 15961.
- Un patrón de Tamaño de ID que denota un tamaño de la tabla (y, por lo tanto, una longitud de bit del Mapa de ID, cuando se utiliza en un IDMPO), que será uno de los patrones definidos en la *Tabla I-5*. El tamaño indicado de la tabla en este campo debe ser menor o igual que el tamaño de la tabla indicado en la tabla de ID seleccionada. El objetivo de este campo es que el decodificador pueda analizar más allá de la Lista de ID o el Mapa de ID, incluso si la Tabla de ID no está disponible para el decodificador.
- Un patrón del Subconjunto de ID de tres bits. La Tabla de ID de Base Primaria del formato de datos registrado, si la utiliza el Objeto Empaquetado actual, siempre se indicará mediante un patrón del Subconjunto de ID codificado de '000'. Sin embargo, también se pueden definir hasta siete Tablas Base Alternativas en el registro (con Tamaños de ID variables), y una opción entre ellas se puede indicar mediante el patrón del Subconjunto codificado. Esta función puede resultar útil para definir subconjuntos específicos de un sector más pequeño o de una aplicación específica de un sistema de datos completo, reduciendo así sustancialmente el tamaño del Mapa de ID codificado.

I.5.3.2 Bits de uso completo o restringido

Cuando se contempla el uso de nuevos registros de la Tabla de ID o registros para sistemas de datos externos, los diseñadores de aplicaciones pueden utilizar una opción de codificación de “uso restringido” que añade cierta sobrecarga a un Objeto Empaquetado, pero a cambio da como resultado un formato que se puede decodificar completamente mediante la recepción de sistemas que no poseen la tabla de ID nueva o externa. Con la excepción de un IDLPO que utiliza el formato predeterminado de Información de Objeto, se codifica un bit de Uso Completo o Restringido inmediatamente después de que se represente cada tabla de ID en la sección Mapa de ID o en la sección Listas de ID de un Paquete de Datos o Directorio

Objeto. En un Objeto Empaquetado con Directorio, este bit siempre se establecerá en '0' y su valor se ignorará. Si un codificador desea utilizar la opción “uso restringido” en un IDLPO, debe anteponer al IDLPO una sección de indicadores de formato que invoque el formato de Información de Objeto no predeterminado.

Si un bit de “Uso Completo o Restringido” es '0', la codificación de las secuencias de datos de la Tabla de ID registrada hace uso completo de la información IDString y FormatString de la Tabla de ID. Si el bit es '1', significa que se ha añadido cierta sobrecarga de codificación a la sección ID Secundario y (en el caso de la compactación de Objetos Empaquetados) a la sección Formato Auxiliar, de modo que un decodificador sin acceso a la tabla puede, sin embargo, producir OID y datos del Objeto Empaquetado según el esquema especificado en *J.4.1*. Específicamente, un bit de uso completo o restringido que sea '1' indica que:

- Para cada Valor de ID codificado, el codificador agregó un indicador EBV-3 a la sección ID Secundario, para indicar cuántos bits de ID Secundario se invocaron con ese Valor de ID. Si el EBV-3 es distinto de cero, siguen los bits de ID Secundario (como indica la entrada de la tabla); pero van seguidos de otro EBV-3, hasta que se haya representado toda la lista de Valores de ID.
- El codificador no aprovechó la información de la columna FormatString de la tabla a la que se hace referencia. En su lugar, correspondiente a cada valor de ID, el codificador insertó un EBV-3 en la sección Formato Auxiliar, indicando el número de longitudes de la cadena de datos discretas, invocadas por el Valor ID (que podría ser más de una debido a las combinaciones y/o los componentes opcionales), seguido del número indicado de longitudes de la cadena. Cada longitud se codifica como si no hubiera FormatString en la tabla de ID. Todos los elementos de datos se codificaron en la subsección A/N de la sección Datos.

I.5.4 Representación de valores de ID en un Objeto empaquetado de la Lista de Valores de ID

Cada Valor de ID se representa dentro de un IDLPO en una lista de campos de bits; el número de campos de bits de la lista se determina a partir del campo NumberOfIDs (consulte Sección *I.5.6.2*). La longitud de cada campo de bit de Valor de ID está en el rango de cuatro a once bits, dependiendo del tamaño del índice de la Tabla Base que representa. En el formato opcional no predeterminado de la sección Información del Objeto de IDLPO, un único Objeto Empaquetado puede contener varias subsecciones de la lista de ID, cada una de las cuales hace referencia a una Tabla de ID diferente. En este formato no predeterminado, cada subsección Lista de ID consta de una subsección Indicador de la Aplicación (que termina las Listas de ID, si comienza con un bit '0'), seguida de un NumberOfID EBV-3, una Lista de ID y un indicador de Uso Completo o Restringido.

I.5.5 Representación de Valores de ID en un Objeto Empaquetado del Mapa de ID

La codificación de un Mapa de ID puede ser más eficaz que la codificación de una lista de Valores de ID, cuando representa una cantidad relativamente grande de Valores de ID (que constituyen más del 10 por ciento, aproximadamente, de las entradas de una Tabla Base grande, o alrededor del 25 por ciento de las entradas de una Tabla Base pequeña). Cuando se codifica en un Mapa de ID, cada Valor de ID se representa mediante su posición correspondiente dentro del Mapa (por ejemplo, el primer bit del Mapa de ID representa el valor de ID “0”, el tercer bit representa el valor de ID “2”, y el último bit representa el valor ID 'n' (correspondiente a la última entrada de una Tabla Base con (n+1) entradas). El valor de cada bit dentro de un Mapa de ID indica si el Valor de ID correspondiente está presente (si el bit es '1') o ausente (si es '0'). Un Mapa de ID siempre se codifica como parte de una estructura de sección de Mapa de ID (consulte *I.9.1*).

I.5.6 Subsección del Apéndice Opcional de la sección Información del Objeto

La función Apéndice del Objeto Empaquetado admite operaciones de edición básicas, en concreto la capacidad de agregar, eliminar o reemplazar elementos de datos individuales en un Objeto Empaquetado previamente escrito, sin necesidad de volver a escribir todo el Objeto Empaquetado. Un Objeto Empaquetado que no contiene una subsección de Apéndice no se puede editar de esta forma y debe volver a escribirse completamente si se requieren cambios.

Una subsección Apéndice consta de un bit de Vínculos Inversos, seguido de un bit Secundario, seguido de uno o dos vínculos EBV-6. Los vínculos de un Objeto Empaquetado de Datos solo se irán a otros objetos empaquetados de datos como adiciones; los vínculos de un Objeto Empaquetado con Directorio solo se irán a otros Objetos Empaquetados con Directorio como adiciones. Se aplican las reglas de estructura de Objeto Empaquetado estándar, con algunas restricciones que se describen en *I.5.6.2*.

El bit de Vínculos Inversos se establecerá de forma idéntica en cada Objeto Empaquetado de la misma “cadena”. El bit de Vínculos Inversos se define de la siguiente manera:

- Si el bit de Vínculos Inversos es '0', cada elemento secundario de esta cadena de Objetos Empaquetados se encuentra en una ubicación de la memoria superior a la de su elemento primario.

El vínculo a un objeto Secundario se codifica como el número de octetos (más uno) que están entre el último octeto del Objeto Empaquetado actual y el primer octeto del objeto Secundario. El vínculo al objeto Primario se codifica como el número de octetos (más uno) que están entre el primer octeto del Objeto Empaquetado principal y el primer octeto del Objeto Empaquetado actual.

- Si el bit de Vínculos Inversos es '1', cada elemento secundario de esta cadena de Objetos Empaquetados se encuentra en una ubicación de la memoria inferior a la de su elemento primario. El vínculo a un objeto Secundario se codifica como el número de octetos (más uno) que están entre el primer octeto del Objeto Empaquetado actual y el primer octeto del objeto Secundario. El vínculo a un objeto principal se codifica como el número de octetos (más uno) que están entre el último octeto del Objeto Empaquetado actual y el primer octeto del objeto principal.

El bit Secundario se define de la siguiente manera:

- Si el bit Secundario es '0', este Objeto Empaquetado es un Objeto Empaquetado "sin matriz" editable (es decir, el primero de una cadena), y en este caso, el bit secundario va seguido inmediatamente de un único enlace EBV-6 al primer Objeto Empaquetado "secundario" que contiene adiciones de edición para el objeto principal.
- Si el bit Secundario es '1', este Objeto Empaquetado es un "secundario" editable de un "principal" editado, y el bit va seguido inmediatamente de un enlace EBV-6 al "principal" y una segunda línea EBV-6 al siguiente Objeto Empaquetado "secundario" que contiene adiciones de edición para el principal.

Un valor vínculo de cero es un puntero Nulo (no existe ningún secundario) y, en un Objeto Empaquetado cuyo bit Secundario es '0', indica que el Objeto Empaquetado es editable, pero que aún no se ha editado. Se proporciona un vínculo al objeto Principal, de modo que un Directorio puede indicar la presencia y ubicación de un Valor de ID en un Objeto Empaquetado del Apéndice. Al mismo tiempo proporciona a un interrogador la capacidad de localizar de forma eficaz los otros Valores de ID que están asociados lógicamente con el Objeto Empaquetado "principal" original. Un valor vínculo cero no es válido como puntero hacia un objeto Principal.

Para permitir espacio para un enlace suficientemente grande, cuando se desconoce la ubicación futura del siguiente "secundario" en el momento en que se codifica el principal, se permite utilizar la forma "redundante" del EBV-6 (por ejemplo, utilizando "100000 000000" para representar un valor de enlace de cero).

I.5.6.1 Lista del Apéndice "EditingOP" (solo en Objetos Empaquetados de la Lista de ID)

Solo en un IDLPO, cada sección Apéndice de un Objeto Empaquetado de la Lista de ID "secundario" contiene un conjunto de bits "EditingOp" codificados inmediatamente después de su último enlace EBV-6. El número de dichos bits se determina a partir de la cantidad de entradas de la lista ID del Objeto Empaquetado de Apéndice. Para cada Valor de ID en esta lista, el bit o los bits EditingOp correspondientes se definen de la manera siguiente:

- '1' significa que se Sustituye el Valor de ID Completo (FQIDV) correspondiente. Una operación de Sustitución tiene el efecto de que los datos originalmente asociados con el FQIDV que coincidan con el FQIDV en este Objeto Empaquetado del Apéndice se ignorarán y se sustituirán lógicamente por los bits de Formato Auxiliar y los datos codificados en este Objeto Empaquetado de Apéndice)
- '00' significa que el FQIDV correspondiente se Elimina pero no se sustituye. En este caso, ni los bits del Formato Auxiliar ni los datos asociados con este Valor de ID se codifican en el Objeto Empaquetado del Apéndice.
- '01' significa que se añade el FQIDV correspondiente (este FQIDV no se codificó previamente ni se eliminó previamente sin reemplazo). En este caso, los bits del Formato Auxiliar y los datos asociados se codificarán en el Objeto Empaquetado del Apéndice.

Nota: Si una aplicación solicita varias operaciones de "edición" a la vez (incluidas algunas operaciones de Eliminación o Sustitución, así como Adiciones), las implementaciones pueden lograr una codificación más eficaz si las adiciones comparten la sobrecarga del apéndice, en lugar de implementarse en un nuevo Objeto Empaquetado.

I.5.6.2 Objetos Empaquetados que contienen una subsección de adición

Un Objeto Empaquetado que contiene una subsección Apéndice; por lo demás, una estructura idéntica a la de otros Objetos Empaquetados. Sin embargo, se aplican las siguientes observaciones:

- Un Objeto Empaquetado "sin matriz" (el primero de una cadena) puede ser un Objeto Empaquetado de la Lista de ID o un Objeto Empaquetado del Mapa de ID (y un IDMPO sin matriz puede ser un IDMPO de datos o con directorio). Cuando un PO "sin matriz" es un directorio, solo se pueden utilizar los IDMPO con directorio como adiciones.

Los bits de Mapa de IDMPO con un Directorio se actualizarán para reflejar correctamente el estado final de la cadena de adiciones y eliminaciones al banco de memoria. No se utiliza un Anexo para el Directorio para realizar este mantenimiento (un anexo al directorio únicamente puede agregar nuevos componentes estructurales, como se describe más adelante en esta sección). Por el contrario, cuando el objeto sin matriz editado es un Objeto Empaquetado de la Lista de ID o un Objeto Empaquetado del Mapa de ID, su Lista de ID o Mapa de ID no se pueden actualizar para reflejar el estado final del Objeto agregado (principales más secundarios).

- Aunque un "secundario" puede ser un Objeto Empaquetado de la Lista de ID y el Mapa de ID, solo un IDLPO puede indicar eliminaciones o cambios en el conjunto actual de Valores de ID completos y datos asociados incorporados en la cadena.
  - Cuando un secundario es un IDMPO, solo se utilizará para agregar (no eliminar ni modificar) información estructural, y no se utilizará para modificar la información existente. En una cadena de Directorios, un IDMPO secundario puede agregar nuevas tablas de ID o puede agregar una nueva sección o subsecciones de AuxMap o puede ampliar una lista de la Tabla de Índice de PO u ObjectOffsets existente. En una cadena de Datos, no se utilizará un IDMPO como Apéndice, excepto para agregar nuevas Tablas de ID.
  - Cuando un secundario es un IDLPO, su lista de ID (seguida de bits "EditingOp") enumera solo aquellos FQIDV que se han eliminado, agregado o reemplazado, en relación con la Lista de ID acumulada de los Objetos anteriores vinculados al mismo.

I.6 Sección Bits de ID secundarios

Los requisitos de diseño de los Objetos Empaquetados incluyen el requisito de que todos los Identificadores del sistema de datos (AI, DI, etc.) codificados en un Objeto Empaquetado puedan reconocerse completamente sin ampliar los datos comprimidos, aunque algunos Valores de ID solo proporcionen un Identificador parcialmente cualificado. Como resultado, si alguno de los Valores de ID invoca bits de ID secundarios, la sección Información de Objeto debe ir seguida de una sección Bits de ID secundarios. Los ejemplos incluyen un campo de cuatro bits para identificar el tercer dígito de un grupo de AI de Logística relacionados.

Los bits de ID secundarios se pueden invocar por varias razones, según sea necesario, para especificar completamente los Identificadores. Por ejemplo, un único Valor de ID de una entrada de la Tabla ID puede especificar una opción entre dos identificadores similares (que requieren un bit codificado para seleccionar uno de los dos ID en el momento de la codificación), o puede especificar una combinación de identificadores obligatorios y opcionales (que requieren un bit codificado para activar o desactivar cada opción). Los mecanismos disponibles se describen en el Anexo J. Todos los campos de bit de ID Secundario resultantes se concatenan en esta sección de bits de ID Secundario, en el mismo orden en que los Valores de ID que los invocaron se enumeraron en el Objeto Empaquetado. Tenga en cuenta que la sección Bits de ID secundarios está definida de forma idéntica, tanto si el Objeto Empaquetado es un IDLPO como un IDMPO, pero no está presente en un IDMPO con Directorio.

I.7 Sección Formato Auxiliar

La sección Formato Auxiliar de un Objeto Empaquetado de Datos codifica la información suplementaria para el proceso de decodificación. Un Objeto Empaquetado con Directorio no contiene una sección Formato Auxiliar. En un Objeto Empaquetado de Datos, la sección Formato Auxiliar comienza con los bits "Parámetro-Compacto", tal como se define en la tabla siguiente.

Table I-7 Patrones de bits Parámetro-Compacto

Patrón de bits	Método de compactación que se usa en este Objeto Empaquetado	Referencia
'1'	Compactación de "objetos empaquetados"	*Consulte I.7.2*
'000'	"Definido por la aplicación", como se define para el método de acceso sin directorio	Consulte I.7.1
'001'	"Compacto", como se define para el método de acceso sin directorio	Consulte I.7.1
'010'	"UTF-8", como se define para el método de acceso sin directorio	Consulte I.7.1
'011bbbb'	("bbbb" estará en el rango de 4..14): reservado para una definición futura	Consulte I.7.1

Si el patrón de bits Parámetro-Compacto es "1", el resto de la sección Formato Auxiliar se codifica, tal como se describe en *I.7.2;* de lo contrario, el resto de la sección del Formato Auxiliar se codifica como se indica en I.7.1.

I.7.1 Compatibilidad con métodos de compactación sin directorio

Si se seleccionó alguno de los métodos de compactación Sin Directorio mediante los bits Parámetro -Compacto, estos bits van seguidos de un patrón de relleno de alineación de bytes que consta de cero o más bits '0' seguidos de un solo bit '1', de modo que el siguiente bit después de '1' se alinea como el bit más significativo del siguiente byte.

Este byte siguiente se define como el primer octeto de una “sección de Datos Sin Directorio”, que se utiliza en lugar de la sección de Datos descrita en I.8. Las cadenas de datos de este Objeto Empaquetado se codifican en el orden indicado por medio de la sección Información del Objeto del Objeto Empaquetado, compactadas exactamente como se describe en el Anexo D de [ISO15962] (Reglas de codificación para el Método de Acceso Sin Directorio), con las dos excepciones siguientes:

- El Identificador de Objetos no está codificado en la “sección Datos Sin Directorio”, porque ya se ha codificado en las secciones Información del Objeto e ID Secundario.
- La versión previa se modifica únicamente en lo que respecta a que los tres bits de Código de Tipo de Compactación sean significativos, y que los otros bits de dicha versión se establezcan en '0'.

Por lo tanto, cada una de las cadenas de datos invocadas a través de la entrada de la Tabla de ID se codifican por separado en la estructura de un conjunto de datos modificada como:

<Versión anterior modificada> <longitud del objeto compactado> <octetos del objeto compactado>

Los <octetos del objeto compactado> se determinan y codifican según los descrito en D.1.1 y D.1.2 de [ISO15962] y la <longitud del objeto compactado> se determina y codifica según lo descrito en D.2 de [ISO15962].

Después del último conjunto de datos, no se codificará un valor de la versión anterior con terminación en cero (el sistema de decodificación reconoce el final de los datos con ObjectLength codificada del Objeto Empaquetado).

I.7.2 Compatibilidad con el método de compactación del objeto empaquetado

Si se seleccionó el método de compactación de Objetos Empaquetados mediante los bits Parámetro -Compacto, estos bits van seguidos de cero o más bits de Formato Auxiliar, según lo invoquen las entradas de la Tabla de ID utilizadas en este Objeto Empaquetado. A continuación, los bits de Formato Auxiliar van seguidos inmediatamente de una sección de Datos que usa el método de compactación de Objetos Empaquetados descrito en I.8.

Una entrada de la Tabla de ID diseñada para su uso con el método de compactación de Objetos Empaquetados puede llamar a varios tipos de información auxiliar más allá de la indicación completa del ID en sí (como campos de bits para indicar una longitud de datos variable, para ayudar al proceso de compactación de datos). Todos estos campos de bits se concatenan en esta parte, en el orden que solicita la Lista o el Mapa de ID. Tenga en cuenta que la sección de Formato Auxiliar se define de forma idéntica, tanto si el Objeto Empaquetado es un IDLPO como si es un IDMPO.

Una entrada de la Tabla de ID invoca los bits de longitud del Formato Auxiliar para todas las entradas que no se especifican como de longitud fija en la tabla (sin embargo, estos bits de longitud no se codifican realmente si corresponden al último elemento de datos codificado en la subsección A/N de un Objeto Empaquetado). Esta información le permite al sistema de decodificación analizar los datos descodificados en cadenas de las longitudes adecuadas. Una entrada codificada de la longitud del Formato Auxiliar utiliza una cantidad variable de bits, que se determina a partir del rango especificado entre las cadenas de datos más cortas y más largas permitidas para el elemento de datos, de la manera siguiente:

- Si se especifica una longitud máxima y el rango especificado (definido como la longitud máxima menos la longitud mínima) es menor que ocho o mayor que 44, las longitudes de este intervalo se codifican en el menor número de bits que puedan expresar longitudes dentro de ese intervalo, y un valor codificado de cero representa la longitud mínima especificada en la cadena de formato. Por ejemplo, si se especifica que el intervalo es de entre tres y seis caracteres, las longitudes se codifican con dos bits y '00' representa una longitud de tres.
- De lo contrario, (incluido el caso de una longitud máxima no especificada), el valor (longitud real - mínimo especificado) se codifica en una cantidad variable de bits, de la manera siguiente:
- Los valores de 0 a 14 (que representan longitudes de 1 a 15, por ejemplo, si la longitud mínima especificada es un carácter) se codifican en cuatro bits
- Los valores de 15 a 29 se codifican en ocho bits (un prefijo de “1111” seguido de cuatro bits que representan valores de 15 (“0000”) a 29 (“1110”)

- Los valores de 30 a 44 se codifican en doce bits (un prefijo de "1111 1111", seguido de cuatro bits que representan valores de 30 ("0000") a 44 ("1110")
- Los valores superiores a 44 se codifican como un prefijo de doce bits de todos los "1", seguido de una indicación EBV-6 de (valor - 44).

Notas:

- Si se especifica un intervalo con límites superiores e inferiores idénticos (es decir, un intervalo de cero), éste se maneja como una longitud fija, no como una longitud variable ni se invoca ningún bit de Formato Auxiliar.
- Si no se especifica un intervalo o tiene límites superiores o inferiores sin especificar, éste se trata como un límite inferior predeterminado de uno y/o un límite superior ilimitado.

I.8 Sección de Datos

Una sección de Datos siempre está presente en un Objeto Empaquetado, excepto en el caso de un Objeto Empaquetado con Directorio o de un Objeto Empaquetado del Apéndice con Directorio (que no codifica elementos de datos), el caso de un Objeto Empaquetado del Apéndice de Datos que únicamente contiene operaciones de Eliminación, y el caso de un Objeto Empaquetado que utiliza la compactación sin directorio (consulte *I.7.1*). Cuando existe una sección de Datos, sigue la sección Información del Objeto (y las secciones ID secundario y Formato Auxiliar, si las hay). Dependiendo de las características de los ID codificados y las cadenas de datos, la sección de Datos puede incluir una o ambas de las dos subsecciones en el orden siguiente: una subsección Numérica de Longitud Conocida, y una subsección Alfanumérica. En los párrafos siguientes se proporcionan descripciones detalladas de cada una de estas subsecciones de la Sección de Datos. Si todas las subsecciones de la sección de Datos se utilizan en un Objeto Empaquetado, el diseño de la sección de Datos es el que se muestra en la tabla siguiente.

Tabla I-8 Estructura máxima de la sección de Datos de Objetos Empaquetados

Subsección Numérica de Longitud Conocida				Subsección Alfanumérica							
				Bits del encabezado A/N				Segmentos de datos binarios			
1er KLN binario	2° KLN binario		Último KLN binario	Bit(s) base no numérico	Bit prefijo, Ejecución(s) prefijo	Bit sufijo, Ejecución sufijo(s)	Mapa de Carac	Num. Binario Ext	Num. Binario Ext	Binario Base 10	Binario No Num.

I.8.1 Subsección Numérica de Longitud Conocida de la sección de Datos

Para cadenas de datos siempre numéricas, la tabla de ID puede indicar un número fijo de dígitos (esta información de longitud fija no está codificada en el Objeto Empaquetado) y/o un número variable de dígitos (en cuyo caso la longitud de la cadena se codificó en la sección Formato Auxiliar, como se describió anteriormente). Cuando se especifica un único elemento de datos en la columna FormatString (consulte *J.2.3*) que contiene una cadena numérica de longitud fija seguida de una cadena de longitud variable; la cadena numérica se codifica en la subsección valores numéricos de longitud conocida y la cadena alfanumérica en la subsección Alfanumérica.

La suma de información de longitud fija (derivada directamente de la tabla de ID) más la información de longitud variable (derivada de bits codificados como se acaba de describir) da como resultado una "entrada de longitud conocida" para cada una de las cadenas siempre numéricas codificadas en el Objeto Empaquetado actual. Cada cadena de datos completamente numérica de un Objeto Empaquetado (si se describe como completamente numérica en la tabla de ID) se codifica al convertir la cadena de dígitos en un único número Binario (hasta 160 bits, que representan un valor binario entre 0 y (10^{48}1)). La figura K-1 del Anexo *K* muestra el número de bits necesarios para representar un número determinado de dígitos. Si una cadena completamente numérica contiene más de 48 dígitos, los primeros 48 se codifican como un grupo de 160 bits, seguidos del siguiente grupo de hasta 48 dígitos, y así sucesivamente. Por último, los valores Binarios de cada cadena de datos completamente numérica del Objeto se concatenan para formar la subsección Numérica de Longitud Conocida.

I.8.2 Subsección alfanumérica de la sección de Datos

La subsección Alfanumérica (A/N), si está presente, codifica todos los datos del Objeto Empaquetado de cualquier cadena de datos que no se haya codificado ya en la subsección Numérica de Longitud Conocida. Si no hay caracteres alfanuméricos para codificar, se omite toda la subsección A/N. La subsección Alfanumérica puede codificar cualquier combinación de dígitos y caracteres ASCII que no sean dígitos o datos de ocho bits. Los caracteres de dígitos de estos datos se codifican por separado, con una eficiencia media de 4,322 bits por dígito o mejor, dependiendo de la secuencia de caracteres.

Los caracteres que no son dígitos se codifican de forma independiente con una eficiencia media que oscila entre 5.91 bits por carácter o mejor (todas las letras mayúsculas), hasta un límite en el peor de los casos de 9 bits por carácter (si la mezcla de caracteres requiere la codificación Base 256 de caracteres no numéricos).

Una subsección Alfanumérica consta de una serie de bits del Encabezado A/N (véase I.8.2.1), seguida de uno a cuatro segmentos Binarios (cada segmento representa datos codificados en una única base numérica, como Base 10 o Base 30, véase I.8.2.4), rellenados, si es necesario, para completar el byte final (Véase I 8.2.5).

I.8.2.1 Bits del Encabezado A/N.

Los Bits del Encabezado A/N se define de la siguiente manera:

- Uno o dos bits Base No Numéricos, como se indica a continuación:
  - '0' indica que se eligió Base 30 para la Base no numérica;
  - '10' indica que se eligió Base 74 para la Base no numérica;
  - '11' indica que se eligió Base 256 para la Base no numérica
- Un solo bit '0' (que indica que no hay ningún Prefijo de Mapa de Caracteres codificado), o un bit '1' seguido de una o más “Ejecuciones” de seis bits de Prefijo, tal como se define en I.8.2.3.
- Un solo bit '0' (que indica que no hay ningún Sufijo de Mapa de Caracteres codificado), o un bit '1' seguido de una o más “Ejecuciones” de seis bits de Sufijo, tal como se define en I.8.2.3.
- Patrón de bits de “Mapa de caracteres” de longitud variable (consulte I.8.2.2), que representa la base de cada uno de los caracteres de datos, si los hubiera, que no fueron contabilizados por un Prefijo o Sufijo.

I.8.2.2 Codificación del mapa de caracteres de base doble

La compactación de la lista ordenada de cadenas de datos alfanuméricos (excluyendo aquellas cadenas de datos ya codificadas en la subsección Numérica de Longitud Conocida) se logra concatenando primero los caracteres de datos en una sola cadena de datos (las longitudes de cadena individuales ya se han registrado en la sección Formato Auxiliar). Cada uno de los caracteres de datos se clasifica como Base 10 (para dígitos numéricos), Base 30 sin valores numéricos (principalmente mayúsculas de la A a la Z), Base 74 sin valores numéricos (que incluye letras mayúsculas y minúsculas y otros caracteres ASCII) o caracteres Base 256. Estos conjuntos de caracteres están completamente definidos en el Anexo K. Además, se puede tener acceso a todos los caracteres del conjunto Base 74 desde la Base 30 con la ayuda de un valor extra de “desplazamiento” (como la mayoría de los 128 caracteres inferiores del conjunto Base 256). Dependiendo del porcentaje relativo de los valores “nativos” de la Base 30 en comparación con otros valores de la cadena de datos, se selecciona una de esas bases como la opción más eficaz para una base no numérica.

A continuación, se graba y codifica la secuencia precisa de caracteres numéricos y no numéricos mediante un patrón de bits de longitud variable, denominado “Mapa de caracteres”, donde cada '0' representa un valor de Base 10 (codificando un dígito) y cada '1' representa un valor para un carácter no numérico (en la base seleccionada). Tenga en cuenta que, (por ejemplo) si se seleccionó la codificación Base 30, cada carácter de datos (que no sean letras mayúsculas ni el carácter de espacio) debe estar representado por un par de valores Base 30, y por lo tanto, cada carácter de datos se representa mediante ***un par*** de bits '1' en el mapa de caracteres.

I.8.2.3 Codificación de la Longitud de Ejecución de Prefijo y Sufijo

Para mejorar la eficacia en los casos en los que la secuencia concatenada incluye secuencias de seis o más valores de la misma base, se prevén representaciones opcionales de la longitud de ejecución de una o más “ejecuciones” de prefijo o sufijo (secuencias de caracteres de una sola base), que pueden reemplazar la primera y/o la última parte del mapa de caracteres. El codificador no debe crear una Ejecución que separe un valor de Desplazamiento de su siguiente valor (desplazado) y, por tanto, una Ejecución siempre representa un número integral de caracteres origen.

Una Representación de Prefijo opcional, si la hay, consta de una o más apariciones de la Ejecución de un Prefijo. Cada Ejecución de un Prefijo consta de un bit de Posición de Ejecución, seguido de dos Bits Base, seguido de tres bits de la Longitud de Ejecución, definidos de la siguiente manera:

- El bit de Posición de Ejecución, si es '0', indica que al menos la Ejecución de un Prefijo más está codificada después de ésta (lo que representa otro conjunto de caracteres origen a la derecha del conjunto actual). El bit de Posición de Ejecución, si es '1', indica que la Ejecución del Prefijo actual es la última (más a la derecha) Ejecución de Prefijo de la subsección A/N.

- El primer bit de base indica una opción de base numérica frente a base no numérica, y el segundo bit de base, si es '1', indica que la base elegida se extiende para incluir caracteres de la base "opuesta". Así, '00' indica una secuencia codificada de longitud de ejecución de valores base 10; '01' indica una secuencia que es principalmente (pero no enteramente) dígitos, codificados en Base 13; '10' indica una secuencia de valores de la base no numérica seleccionada anteriormente en el encabezado A/N, y '11' indica una secuencia de valores principalmente de esa base no numérica, pero extendida para incluir también caracteres numéricos. Cabe mencionar una excepción: si la base no numérica que se seleccionó en el encabezado A/N es Base 256, entonces, la versión "extendida" se define como Base 40.

- El valor de la Longitud de Ejecución de 3 bits supone una ejecución mínima utilizable de seis caracteres de la misma base, y el valor de longitud se divide aún más entre 2. Por lo tanto, los posibles valores de Longitud de Ejecución de 3 bits de 0, 1, 2, ... 7 indican una Ejecución de 6, 8, 10, ... 20 caracteres de la misma base. Tenga en cuenta que un valor de carácter "extraño" final al final de una secuencia de la misma base debe representarse agregando un bit al Mapa de Caracteres.

Una Representación del Sufijo opcional, si la hay, es una serie de una o más Ejecuciones de Sufijo, cada una con el mismo formato que la Ejecución de Prefijo que se acaba de describir. De conformidad con esa descripción, tenga en cuenta que el bit de Posición de la Ejecución, si es '1', indica que la Ejecución de Sufijo actual es la última Ejecución de Sufijo (más a la derecha) de la subsección A/N y, por lo tanto, cualquier Ejecución de Sufijo anterior representaba caracteres de origen a la izquierda de esta Ejecución de Sufijo final.

I.8.2.4 Codificación en segmentos binarios

Inmediatamente después del último bit del mapa de caracteres, se codifican hasta cuatro números binarios, cada uno de los cuales representa todos los caracteres que se codificaron en un sistema base único. Primero, se codifica una secuencia base de 13 bits (si una o más Ejecuciones de prefijo o sufijo requieren codificación de base 13). Si está presente, esta secuencia de bits representa directamente el número binario resultante de codificar la secuencia combinada de todos los caracteres de prefijo y sufijo (en ese orden) clasificados como Base 13 (ignorando cualquier carácter de intervención no clasificado de esta manera) como un valor único, o en otras palabras, de aplicar una base 13 a la conversión binaria. El número de bits para codificar en esta secuencia se determina directamente a partir del número de valores de base 13 que se representan, como lo requiere la suma de las longitudes de las ejecuciones de prefijo y sufijo para las secuencias de base 13. El número de bits, para un número dado de valores de Base 13, se determina a partir de la Figura del Anexo *K*. Posteriormente, se codifica un segmento de Base extendida-no numérica (Base-40 o Base 84) de forma similar (si una o más Ejecuciones de prefijo o sufijo requieren codificación extendida-no numérica).

A continuación, se codifica un segmento binario de Base 10 que representa directamente el número binario resultante de codificar la secuencia de dígitos en el prefijo o mapa de caracteres o sufijo (ignorando cualquier carácter de intervención que no sea un dígito) como un valor único, o en otras palabras, aplicando una base 10 a la conversión binaria. El número de bits a codificar en esta secuencia se determina directamente a partir del número de dígitos que se representan, como se muestra en el Anexo *K*.

Inmediatamente después del último bit de la secuencia de bits de Base-10 (si corresponde), se codifica una secuencia de bits no numérica (Base 30, Base 74 o Base 256) (si el mapa de caracteres indica al menos un carácter no numérico). Esta secuencia de bits representa el número binario resultante de una conversión de base 30 a binaria (o una conversión de base 74 a binaria, o una transferencia directa de valores de Base-256) de la secuencia de caracteres que no son dígitos en los datos (ignorando cualquier dígito de intervención). Nuevamente, el número de bits codificados se determina directamente a partir del número de valores no numéricos que se representan, como se muestra en el Anexo *K*. Considere que, si se seleccionó la Base 256 como base no numérica, el codificador es libre de clasificar y codificar cada dígito ya sea como Base 10 o como Base 256 (la Base 10 será más eficiente, a menos que la capacidad de aprovechar un prefijo o sufijo largo lo supere).

Tenga en cuenta que una subsección alfanumérica termina con varios campos de bits de longitud variable (el mapa de caracteres y una o más secciones binarias (que representan los valores binarios numéricos y no numéricos). Asimismo, tenga en cuenta que ninguna de las longitudes de estos tres campos de bits de longitud variable está codificada explícitamente (aunque uno o dos segmentos binarios de base extendida también pueden estar presentes, estos tienen longitudes conocidas, determinadas a partir de ejecuciones de prefijo o sufijo).

Para determinar los límites entre estos tres campos de longitud variable, el decodificador necesita implementar un procedimiento, utilizando el conocimiento del número restante de bits de datos, para analizar correctamente la subsección alfanumérica. En el Anexo *M*. se describe un ejemplo de dicho procedimiento.

I.8.2.5 Relleno del último byte

El último bit (menos significativo) del segmento binario final es también el último bit significativo del objeto empaquetado. Si quedan posiciones de bit restantes en el último byte para rellenar con bits de relleno, el bit de relleno más significativo se pondrá a "1" y los bits de relleno menos significativos restantes se pondrán en "0". El decodificador puede determinar el número total de bits que no son de relleno en un objeto empaquetado examinando la sección de longitud del objeto empaquetado (y si el bit indicador de relleno de esa sección es "1", examinando también el último byte del objeto empaquetado).

I.9 Opciones de codificación de mapas de ID y directorios

Un mapa de ID puede ser más eficaz que una lista de valores de ID cuando se codifica un número relativamente grande de valores de ID. Además, una representación de mapa de ID es ventajosa para su uso en un objeto empaquetado de directorio. El mapa de ID en sí (la primera subsección principal de cada sección de mapa de ID) está estructurado de manera idéntica ya sea en un IDMPO de datos o de directorio, pero la sección Mapa de ID de un IDMPO de directorio contiene subsecciones opcionales adicionales. En la siguiente sección se describe la estructura de una sección de mapa de ID, que contiene uno o más mapas de ID, explicada en función de su uso en un IDMPO de datos; las secciones siguientes explican los elementos estructurales agregados en un directorio IDMPO.

I.9.1 Estructura de la sección del mapa de ID

Un IDMPO representa los valores de identificación mediante una estructura llamada sección de mapa de ID, que contiene uno o más mapas de identificación. Cada valor de ID codificado en un IDMPO de datos se representa como un bit "1" dentro de un campo de bits del mapa de ID, cuya longitud fija es igual al número de entradas en la tabla base correspondiente. Por el contrario, cada "0" en el campo de mapa de ID indica la ausencia del valor de ID correspondiente. Dado que el número total de bits '1' dentro del campo de mapa de ID es igual al número de valores de ID que se representan, no se codifica ningún campo NumberOfIDs explícito. Para implementar el rango de funcionalidad que hace posible esta representación, la Sección de mapa de ID contiene elementos distintos del mapa de ID en sí. Si está presente, la sección de mapa de ID opcional sigue inmediatamente el patrón inicial que indica un IDMPO (como se describe en *I.4.2*), y contiene los siguientes elementos en el orden que se indica a continuación:

- Una subsección de indicador de aplicación (vea *I.5.3.1*)
- un campo de bits del mapa de ID (cuya longitud se determina a partir del tamaño de ID en el indicador de aplicación)
- un bit de uso completo/restringido (vea *I.5.3.2*)
- (la secuencia anterior forma un mapa de ID que, opcionalmente, puede repetirse varias veces)
- un bit indicador de datos/directorio,
- una sección AuxMap opcional (nunca presente en un IDMPO de datos), y
- Indicador (es) de cierre, que consta (n) de un bit de "Indicador de apéndice". Si es "1", entonces una subsección de apéndice está presente al final de la sección de Información del objeto (después de la Información de longitud del objeto).

Estos elementos, que se muestran en la siguiente tabla como una estructura máxima (cada elemento está presente), se describen en cada una de las siguientes subsecciones.

Tabla I-9 Sección de mapa de ID

Primer mapa de ID		Mapa (s) de ID adicional (es) opcional (es)		Indicador de aplicación nula	Datos/ Directorio	(Si se elige el directorio) Opcional	Bit (s) de indicador de cierre
Indicador de aplicación	Campo de bits de mapa de ID (termina con el bit F/R)	Indicador de aplicación	Campo de mapa de ID (termina con el bit F/R)	(bit cero único)	Bit indicador	Sección AuxMap	
Consulte *I.5.3.1*	Consulte *I.9.1.1* y *I.5.3.2*	Igual que el anterior	Igual que el anterior	Consulte *I.5.3.1*		Vea Tabla I-12	Bit indicador de apéndice

Cuando se codifica una sección de mapa de ID, siempre va seguida de una Longitud de objeto y un Indicador de relleno, y opcionalmente seguida de una subsección de apéndice (todas como se han definido previamente), y posteriormente puede ir seguida de cualquiera de las otras secciones definidas para objetos empaquetados, excepto que un IDMPO de directorio no incluirá una sección de datos.

I.9.1.1 Mapa de ID y campo de bits del mapa de ID

Un mapa de ID generalmente consta de un indicador de aplicación seguido de un campo de bits del mapa de ID, que termina con un bit de uso completo/restringido. Un campo de bits del mapa de ID consta de un solo bit indicador "MapPresent", por ende (si MapPresent es "1") un número de bits igual a la longitud determinada a partir del patrón de tamaño de ID dentro del indicador de aplicación, más uno (el bit de uso completo/restringido). El campo de bits del mapa de ID indica la presencia o ausencia de elementos de datos codificados correspondientes a entradas en una tabla base primaria o alternativa registrada específica. La elección de la tabla base se indica mediante la combinación codificada de DSFID y el patrón de indicador de aplicación que precede al campo de bits del mapa de ID. El MSB del campo de bits del mapa de ID corresponde al valor de ID 0 en la tabla base, el siguiente bit corresponde al valor de ID 1, y así sucesivamente.

En el campo de bits del mapa de ID de un objeto empaquetado de datos, cada bit "1" indica que este objeto empaquetado contiene una ocurrencia codificada del elemento de datos correspondiente a una entrada en la tabla base registrada asociada con este mapa de ID. Tenga en cuenta que la entrada codificada válida se puede encontrar en el primer objeto empaquetado ("sin matriz") de la cadena (la que contiene el mapa de ID) o en un IDLPO de apéndice de esa cadena. Además, tenga en cuenta que una o más entradas de datos pueden estar codificadas en un IDMPO, pero marcadas como "inválidas" (mediante una entrada de eliminación en un IDLPO del apéndice).

Un mapa de ID no debe corresponder a una tabla de ID secundaria en lugar de una tabla de ID base. Tenga en cuenta que los elementos de datos codificados en un IDMPO de datos "sin matriz" aparecerán en el mismo orden relativo en el que se enumeran en la tabla base asociada. Sin embargo, se pueden agregar elementos de datos adicionales "fuera de servicio" a un IDMPO de datos existente agregando un IDLPO del apéndice al Objeto.

Un mapa de ID no puede indicar un número específico de instancias (mayor que uno) del mismo valor de ID, y esto aparentemente implicaría que solo una instancia de datos que usa un valor de ID dado puede codificarse en un IDMPO de datos. No obstante, el método del mapa de ID debe admitir el caso en el que dos o más elementos de datos codificados son de la misma "clase" de identificador (y, por lo tanto, comparten el mismo valor de ID). Los siguientes mecanismos abordan esta necesidad:

- Otro elemento de datos de la misma clase se puede codificar en un IDLPO del apéndice del IDMPO. Puede haber varias apariciones del mismo valor de ID en una lista de ID, cada una asociada con diferentes valores codificados de los bits de ID secundarios.
- Una serie de dos o más instancias codificadas de la misma "clase" se puede indicar de manera eficiente mediante una única instancia de un valor de ID (o, de manera equivalente, mediante un solo bit de mapa de ID), si la entrada correspondiente de la tabla base define un bit de "repetición" (vea *J.2.2*).

Una sección de mapa de ID puede contener varios mapas de identificación; una sección de indicador de aplicación nula (con su bit AppIndicatorPresent establecido en "0") finaliza la lista de mapas de ID.

I.9.1.2 Bits indicadores de datos/directorio y AuxMap

Un bit indicador de datos/directorio siempre se codifica inmediatamente después del último mapa de ID. Por definición, un IDMPO de datos tiene su bit de datos/directorio establecido en "0", y un IDMPO de directorio tiene su bit de datos/directorio establecido en "1".Si el bit de datos/directorio se establece en "1", tiene inmediatamente después un bit indicador de AuxMap que, si es "1", indica que sigue inmediatamente una sección opcional de AuxMap.
Bit (s) de indicadores de cierre

La sección mapa de ID termina con un sol indicador de cierre:

- El bit final de los indicadores de cierre es un bit indicador de apéndice que, si es "1", indica que hay una subsección de apéndice opcional codificada al final de la sección de Información del objeto del objeto empaquetado. Si está presente, la subsección de apéndice es tal cual se describe en la Sección *1.5.6*.

I.9.2 Objetos empaquetados de directorio

Un "Objeto empaquetado de directorio" es un IDMPO cuyo bit de directorio se establece en "1". Su única diferencia inherente de un IDMPO de datos es que no contiene ningún elemento de datos codificados. Sin embargo, los mecanismos adicionales y las consideraciones de uso se aplican únicamente a un objeto empaquetado de directorio, los cuales se describen en las siguientes subsecciones.

I.9.2.1 Mapas de ID en un directorio IDMPO

Aunque la estructura de un mapa de ID es idéntica ya sea en un IDMPO de datos o de directorio, la semántica de la estructura es algo diferente. En el campo de bits del mapa de ID de un objeto empaquetado de directorio, cada bit "1" indica que un objeto empaquetado de datos en el mismo banco de memoria del portador de datos contiene un elemento de datos válido asociado con la entrada correspondiente en la tabla base especificada para este mapa de ID. Opcionalmente, un objeto empaquetado de directorio puede indicar además ***qué*** objeto empaquetado contiene cada elemento de datos (consulte la descripción de la sección AuxMap opcional a continuación).

Considere que, a diferencia de un IDMPO de datos, no existe una correlación requerida entre el orden de los bits en un mapa de ID de directorio y el orden en el que estos elementos de datos se codifican posteriormente en la memoria dentro de una secuencia de objetos empaquetados de datos.

I.9.2.2 Sección AuxMap opcional (únicamente IDMPO de directorio)

Una sección AuxMap permite opcionalmente que el mapa de ID de un directorio IDMPO indique no solo la presencia/ausencia de todos los elementos de datos en este banco de memoria de la etiqueta, sino también qué objeto empaquetado codifica cada elemento de datos. Si el bit indicador AuxMap es "1", se codificará una sección AuxMap inmediatamente después de este bit. Si está codificada, la sección AuxMap contendrá un campo de índice de PO para cada uno de los mapas de ID que preceden a esta sección. Después del último campo de índice de PO, la sección AuxMap puede codificar opcionalmente una lista ObjectOffsets, donde cada ObjectOffset generalmente indica el número de bytes desde el inicio del objeto empaquetado anterior hasta el comienzo del siguiente objeto empaquetado. Esta estructura de AuxMap se muestra (para un IDMPO de ejemplo con dos mapas de ID) en la siguiente tabla.

Tabla I-10 Estructura de la sección AuxMap opcional

Campo de índice de PO para el primer mapa de ID		Campo de índice de PO para el segundo mapa de ID		Bit presente de desplazamientos de objetos	Subsección de desplazamientos de objetos opcional				
Longitud de índice PO	Tabla de índice PO	Longitud de índice PO	Tabla de índice PO		Multiplicador de desplazamientos de objetos	Desplazamiento de objeto1 (EBV6)	Desplazamiento de objeto2 (EBV6)	…	Desplazamiento de objetoN (EBV6)

Cada campo de índice de PO tiene la siguiente estructura y semántica:

- Un campo POindexLength de tres bits, que indica el número de bits de índice codificados para cada entrada en la tabla de índice de PO que sigue inmediatamente a este campo (a menos que la longitud de POIndex sea "000", lo que significa que no sigue ninguna tabla de índice de PO).
- Una tabla de índice de PO, que consta de una matriz de bits, un bit (o grupo de bits, según el POIndexLength) por cada bit en el mapa de ID correspondiente de este objeto empaquetado de directorio. Una entrada de la tabla de índice de PO (es decir, un "índice de PO") indica (por orden relativo) qué objeto empaquetado contiene el elemento de datos indicado por el bit "1" correspondiente en el mapa de ID. Si un bit de mapa de ID es "0", la entrada correspondiente de la tabla de índice de PO está presente, pero se ignora su contenido.

- A cada objeto empaquetado se le asigna un valor de índice en secuencia, sin importar si es un objeto empaquetado "sin matriz" o "secundario" de otro objeto empaquetado, o si es un objeto empaquetado de datos o directorio.
- Si el índice de PO se encuentra dentro de la primera tabla de índice de PO (para el mapa de ID asociado) de la "cadena" del directorio, entonces:
  - un índice de PO de cero se refiere al primer objeto empaquetado en la memoria,
  - un valor de uno se refiere al siguiente objeto empaquetado en la memoria, y así sucesivamente
  - un valor de *m*, donde *m* es el valor más grande que se puede codificar en el índice de PO (dado el número de bits por índice que se estableció en POindexLength), indica un objeto empaquetado cuyo índice relativo (posición en la memoria) es *m* o más alto. Esta definición permite que los objetos empaquetados superiores a *m* se indexen en un objeto empaquetado del directorio del apéndice, como se describe a continuación. Si no existe ningún apéndice, entonces la posición precisa es *m* o una posición indeterminada mayor que *m*.
- Si el Índice de PO no está dentro de la primera tabla de índice de PO de la cadena de directorios para el mapa de ID asociado (es decir, está en un IDMPO del apéndice), entonces:
  - un índice de PO de cero indica que una tabla de índice de PO anterior de la cadena proporcionó la información del índice,
  - un índice de PO de *n* (*n*> 0) se refiere al *enésimo* objeto empaquetado por encima del valor de índice más alto disponible en el PO del directorio primario inmediato; por ejemplo, si el valor de índice máximo en el PO del directorio primario inmediato se refiere al número de PO "3 o mayor", entonces un índice de PO de 1 en este apéndice se refiere al número de PO 4.
  - Un índice de PO de *m* (como se define arriba) indica de manera similar un objeto empaquetado cuya posición es la *emésima* posición, o *más alta*, que el límite de la tabla anterior en la cadena.
- Si la instancia válida de un valor de ID está en un objeto empaquetado de apéndice, una implementación puede optar por establecer un índice de PO para que apunte directamente a ese apéndice o, en su lugar, puede continuar apuntando al objeto empaquetado en la cadena que originalmente contenía el valor de ID.
  NOTA: El primer enfoque a veces conduce a una búsqueda más rápida; el segundo a veces conduce a actualizaciones de directorio más rápidas.

Después del último campo de índice de PO, la sección AuxMap termina con (como mínimo) un solo bit "ObjectOffsets Existentes". Un valor "0" de este bit indica que no se codifica ninguna subsección ObjectOffsets. Si, en cambio, este bit es un "1", presenta inmediatamente después una subsección ObjectOffsets, que contiene una lista de "desplazamientos" EBV-6 (el número de octetos entre el comienzo de un objeto empaquetado y el comienzo del siguiente objeto empaquetado). Si está presente, la subsección ObjectOffsets consta de un ObjectOffsetsMultiplier seguido de una lista de desplazamientos de objetos, definida de la siguiente manera:

- Un ObjectOffsetsMultiplier EBV-6, cuyo valor, cuando se multiplica por 6, establece el número total de bits reservados para toda la lista ObjectOffsets. Se debe seleccionar el valor de este multiplicador para que, idealmente, dé como resultado un almacenamiento suficiente para contener las compensaciones para el número máximo de objetos empaquetados que se pueden indexar mediante la tabla de índice de PO de este objeto empaquetado de directorio (dado el valor en el campo POIndexLength, y dado algunos valores estimados tamaño medio para esos objetos empaquetados).
- un campo de tamaño fijo que contiene una lista de ObjectOffsets EBV-6. El tamaño de este campo es exactamente el número de bits calculado a partir de ObjectOffsetsMultiplier. El primer ObjectOffset representa el inicio del segundo objeto empaquetado en la memoria, en relación con el primer octeto de memoria (sería poco beneficioso reservar espacio adicional para almacenar el desplazamiento del *primer* objeto empaquetado). Cada ObjectOffset sucesivo indica el inicio del siguiente objeto empaquetado (en relación con el ObjectOffset anterior en la lista), y el ObjectOffset final en la lista apunta al patrón de terminación en ceros donde se puede escribir el *siguiente* objeto empaquetado. Se utilizará un desplazamiento de cero inválido (patrón EBV-6 "000000") para terminar la lista ObjectOffset. Si el espacio de almacenamiento reservado está completamente ocupado, no es necesario que incluya este patrón de terminación.
- En aplicaciones en las que la longitud promedio del objeto empaquetado es difícil de predecir, el espacio de almacenamiento reservado de ObjectOffset a veces puede resultar insuficiente. En este caso, se puede agregar un objeto empaquetado del apéndice al objeto empaquetado del directorio. Este objeto empaquetado del directorio del apéndice puede contener subsecciones nulas para todos menos su subsección ObjectOffsets. De forma alternativa, si se prevé que la capacidad de la tabla del índice de PO también se superará eventualmente, el objeto empaquetado del apéndice también puede contener uno o más campos de índice de PO no nulos. Tenga en cuenta que en una instancia determinada de una sección AuxMap, una tabla de índice de PO o una subsección ObjectOffsets pueden ser las primeras en exceder su capacidad.

Por lo tanto, la primera posición a la que hace referencia una lista ObjectOffsets en un objeto empaquetado del apéndice no necesita coincidir con la primera posición a la que hace referencia la tabla de índice de PO de ese mismo apéndice. Específicamente, en un objeto empaquetado del apéndice, el primer ObjectOffset enumerado es un desplazamiento referenciado al último ObjectOffset en la lista del objeto empaquetado de directorio "principal".

I.9.2.3 Uso como directorio de presencia/ausencia

En muchas aplicaciones, un interrogador puede optar por leer el contenido completo de cualquier soporte de datos que contenga uno o más elementos de datos "objetivo" de interés. En dichas aplicaciones, la información de posición de esos elementos de datos dentro de la memoria no es necesaria durante las operaciones de lectura inicial; solo se necesita una indicación de presencia/ausencia en esta etapa de procesamiento. Un mapa de ID puede formar un directorio de presencia/ausencia particularmente eficiente para indicar el contenido de un soporte de datos en tales aplicaciones. Una estructura de directorio completa codifica el desplazamiento o la dirección (localización de la memoria) de cada elemento de datos dentro del soporte de datos, lo que requiere la escritura de una gran cantidad de bits (por lo general, 32 bits o más por elemento de datos). Inevitablemente, este enfoque también requiere leer una gran cantidad de bits en el aire, solo para determinar si un identificador de interés está presente en una etiqueta en particular. Al contrario, cuando solo se necesita información de presencia/ausencia, el uso de un mapa de ID transmite la misma información usando solo un bit por elemento de datos definido en el sistema de datos. El mapa de ID completo se puede representar normalmente en 128 bits o menos, y permanece del mismo tamaño a medida que se escriben más elementos de datos en la etiqueta.

Un objeto empaquetado de "Directorio de presencia/ausencia" se define como un IDMPO de directorio que no contiene un índice de PO y, por lo tanto, no proporciona información codificada sobre dónde residen los elementos de datos individuales dentro del soporte de datos. Un directorio de presencia/ausencia se puede convertir en un objeto empaquetado de "directorio indexado" (vea I.9.2.4) agregando un índice de PO en un objeto empaquetado del apéndice, como "secundario" del objeto empaquetado de presencia/ausencia.

I.9.2.4 Uso como directorio indexado

En muchas aplicaciones que involucran memorias grandes, un interrogador puede optar por leer una sección del directorio que cubra el contenido completo de la memoria y luego emitir lecturas posteriores para buscar los elementos de datos "objetivo" de interés. En estas aplicaciones, la información de posición de esos elementos de datos dentro de la memoria es importante, pero si se agregan muchos elementos de datos a una memoria grande con el tiempo, el directorio en sí puede crecer hasta un tamaño no deseado.

Un mapa de ID, utilizado junto con un AuxMap que contiene un índice de PO, puede formar un "directorio indexado" particularmente eficaz para indicar el contenido de una etiqueta RFID y también sus localizaciones aproximadas. A diferencia de una estructura de directorio de etiqueta completa, que codifica el desplazamiento o la dirección (localización de la memoria) de cada elemento de datos dentro del soporte de datos, un directorio indexado codifica una pequeña posición relativa o índice que indica qué objeto empaquetado contiene cada elemento de datos. Un diseñador de aplicaciones puede optar por codificar también las localizaciones de cada objeto empaquetado en una subsección opcional ObjectOffsets como se describe anteriormente, de modo que un sistema de decodificación, al leer solo el directorio indexado, pueda calcular las direcciones de inicio de todos los objetos empaquetados en la memoria.

La utilidad de un mapa de ID utilizado de esta manera se ve reforzada por la regla de la mayoría de los sistemas de datos de que un identificador dado únicamente puede aparecer una vez dentro de un único soporte de datos. Esta regla, cuando se utiliza un directorio indexado con la codificación de objetos empaquetados de los datos en los objetos posteriores, puede proporcionar un acceso aleatorio casi completo para leer datos utilizando relativamente pocos bits de directorio. Por ejemplo, un directorio de mapa de ID (un bit por ID definido) se puede asociar con una matriz de "índice de PO" adicional de AuxMap (usando, por ejemplo, tres bits por ID definido). Con esta disposición, un interrogador leería el objeto empaquetado en el directorio y examinaría su mapa de ID para determinar si el elemento de datos deseado estaba presente en la etiqueta. Si es así, examinaría los 3 bits del "índice de PO" correspondientes a ese elemento de datos, para determinar cuál de los primeros 8 objetos empaquetados en la etiqueta contiene el elemento de datos deseado. Si se codificó una subsección de ObjectOffsets opcional, el interrogador puede calcular la dirección inicial del objeto empaquetado deseado directamente; de lo contrario, el interrogador puede realizar operaciones de lectura sucesivas para obtener el objeto empaquetado deseado.

J Tablas de ID de objetos empaquetados

J.1 Estructura del archivo de registro del formato de datos de los objetos empaquetados

Un archivo de formato de datos registrado de objetos empaquetados consta de una serie de “líneas de palabras clave” y una o más tablas de identificación. Las líneas en blanco pueden aparecer en cualquier lugar dentro de un archivo de formato de datos y deben ignorarse. Además, cualquier línea puede terminar con columnas en blanco adicionales, que también se ignoran.

- Una línea de palabra clave consta de una palabra clave (que siempre comienza con “K-”) seguida de un signo de igual y una secuencia de caracteres, que asigna un valor a esa palabra clave. Puede haber cero o más caracteres de espacio en cualquier lado del signo de igual. Algunas líneas de palabras clave aparecerán solo una vez, en la parte superior del archivo de registro, y otras pueden aparecer varias veces, una para cada tabla de ID en el archivo.
- Una tabla de ID enumera una serie de valores de ID (como se define en *I.5.3*). Cada fila de una tabla de ID contiene un único valor de ID (en una columna "IDvalue" obligatoria), y las columnas adicionales pueden asociar ID de objeto (OID), secuencias de ID, secuencias de formato y otra información con ese valor de ID. Un archivo de registro siempre incluye una única Tabla de ID de Base “Principal”, cero o más Tablas de ID de Base “Alternativa” y también puede incluir una o más Tablas de ID Secundarias (que se referencian por una o más entradas de la Tabla de ID base).

Para ilustrar el formato de archivo, en la Figura J-1 se muestra un registro de sistema de datos hipotético. En este sistema de datos hipotético, cada valor de ID se asocia con uno o más OID y las secuencias de ID correspondientes. Las siguientes subsecciones explican la sintaxis que se muestra en la figura.

Figura J-1 Archivo de registro de formato de datos hipotéticos

K-texto = Datos hipotéticos
Formato 100

K-Versión =
1.0

K-TableID = F100B0

K-RootOID =
urn:oid:1.0.12345.100

K-IDsize =
16

IDvalue	OIDs	IDstring	Explicación	FormatString
0	99	1Z	El ID heredado "1Z"	14n
1	9%x30-33	7%x42-45	corresponde al OID 99, se le asigna el IDval 0 Un OID en el rango 90..93, Correspondiente a ID 7B..7E	1*8an
2	(10)(20)(25)(37)	(A)(B)(C)(D)	un conjunto de ID de uso común	(1n)(2n)(3n)(4n)
3	26/27	1A/2B	Se codifica 1A o 2B, pero no ambos	10n / 20n
4	(30) [31]	(2A) [3B]	2A siempre está codificado, seguido opcionalmente por 3B	(11n) [1*20n]
5	(40/41/42) (53) [55]	(4A/4B/4C) (5D) [5E]	Uno de A/B/C está codificado, luego D y, opcionalmente, E	(1n/2n/3n) (4n) [5n]
6	(60/61/(64)[66])	(6A /6B / (6C) [6D])	Selecciones, una de las cuales incluye una opción	(1n / 2n / (3n][4n])

K-TableEnd = F100B0

J.1.1 Sección de encabezado de archivo

Las líneas de palabras clave en el encabezado del archivo (la primera parte de cada archivo de registro) pueden aparecer en cualquier orden y son las siguientes:

- **(Obligatorio) K-Versión = nn.nn**, que asigna el organismo de registro, para garantizar que cualquier revisión futura de su registro esté claramente etiquetada.
- **(Opcional) K-Interpretación = secuencia**, donde el argumento "secuencia" será uno de los siguientes: "ISO-646", "UTF-8", "ECI-nnnnnn" (donde nnnnnn es un número ECI registrado de seis dígitos), ISO-8859-nn o "SIN ESPECIFICAR". La interpretación predeterminada es "SIN ESPECIFICAR". Esta línea de palabras clave permite colocar interpretaciones no predeterminadas en los octetos de secuencias de datos que se decodifican a partir de objetos empaquetados.
- **(Opcional) K-ISO15434 = nn**, donde "nn" representa un indicador de formato (un identificador numérico de dos dígitos) como se define en ISO/IEC 15434. Esta línea de palabras clave permite que los sistemas receptores representen opcionalmente un objeto empaquetado decodificado como un mensaje ISO/IEC 15434 totalmente compatible. No hay un valor predeterminado para esta línea de palabras clave.
- **(Opcional) K-AppPunc = nn**, donde nn representa (en decimales) el valor de octeto de un carácter ASCII que se usa comúnmente para la puntuación en esta aplicación. Si esta línea de palabras clave no está presente, el carácter de puntuación de la aplicación predeterminado es el guion.

Además, h puede incluirse usando la línea de asignación de palabra clave opcional "K-texto = secuencia", y puede aparecer cero o más veces dentro de un encabezado de archivo o encabezado de tabla, pero no en el cuerpo de una tabla de ID.

J.1.2 Sección de encabezado de tabla

Una o más secciones de encabezado de tabla (cada una presenta una tabla de ID) se encuentran después de una sección de encabezado de archivo. Cada encabezado de tabla comienza con una línea de palabras clave K-TableID, seguida de una serie de líneas de palabras clave adicionales obligatorias y opcionales (que pueden aparecer en cualquier orden), como se indica a continuación:

- **(Obligatorio) K-TableID = FnnXnn**, donde **Fnn** representa el número de formato de datos asignado por ISO (donde "nn" representa uno o más dígitos decimales), y Xnn (donde "X" es "B" o "S") es un ID de tabla asignado por el registrante para cada tabla de ID en el archivo. La primera tabla de ID siempre será la tabla de ID de base primaria del registro, con una ID de tabla de "B0". Se pueden incluir hasta siete tablas de ID de base "alternativa" adicionales, con ID de tabla "Bnn" secuenciales más altas. Pueden incluirse tablas de ID secundarias, con ID de tabla secuenciales de la forma "Snn".
- **(Obligatorio) K-IDsize = nn.** Para una tabla de ID base, el valor **nn** será uno de los valores de la columna "Número máximo de entradas de tabla" de la Tabla I 5-5. Para una tabla de ID secundaria, el valor **nn** será una potencia de dos (incluso si no está presente en la Tabla I 5-5).
- **(Opcional) K-RootOID = urn: oid: i.j.k.ff** donde:
  - **I, j y k** son los arcos principales de OID (tantos arcos como sea necesario) y
  - **ff** es el último arco del OID raíz (por lo general, el número de formato de datos registrado)

Si la palabra clave K-RootOID no está presente, entonces el OID raíz predeterminado es:

- **urn:oid:1.0.15961.ff**, donde "ff" es el número de formato de datos registrado
- **Otras líneas de palabras clave opcionales:** para anular los valores predeterminados a nivel de archivo (para establecer diferentes valores para una tabla en particular), un encabezado de tabla puede invocar una o más de las líneas de palabras clave opcionales enumeradas en la sección Encabezado de archivo.

El final de la sección Encabezado de la tabla es la primera línea que no está en blanco y que no comienza con una palabra clave. Esta primera línea que no está en blanco incluirá los títulos de cada columna en la Tabla de ID que sigue inmediatamente a esta línea; los títulos de las columnas distinguen entre mayúsculas y minúsculas.

Una tabla de ID de base alternativa, si está presente, tiene un formato idéntico a la tabla de ID de base principal (pero generalmente representa una selección más pequeña de identificadores, destinados a una aplicación específica).

Una tabla de ID secundaria se puede invocar mediante una palabra clave en la columna **OID** de una tabla base. Una tabla de ID secundaria es equivalente a una sola lista de selección (vea *J.3*) para un valor de ID único de una tabla de ID base (excepto que una tabla secundaria usa K-Idsize para definir explícitamente el número de bits de ID secundaria por ID); la columna IDvalue de una tabla secundaria enumera el valor del patrón de bits de ID secundaria correspondiente para cada fila en la tabla secundaria. Una entrada de **OID** en una tabla de ID secundaria no contendrá en sí misma una lista de selección ni invocará otra tabla de ID secundaria.

J.1.3 Sección de la tabla de ID

Cada tabla de ID consta de una serie de una o más filas, cada fila incluye una columna obligatoria "IDvalue", varias columnas opcionales definidas (como "OID", "IDstring" y "FormatString") y cualquier cantidad de columnas informativas (como la columna "Explicación" en el ejemplo hipotético que se muestra arriba).

Cada tabla de ID termina con una línea de palabras clave obligatoria del formulario:

- **K-TableEnd = FnnXnn**, donde **FnnXnn** coincidirá con la línea de palabras clave **K-TableID** anterior que introdujo la tabla.

La sintaxis y los requisitos de todas las columnas obligatorias y opcionales serán los que se describen en J.2.

J.2 Columnas de la tabla de ID obligatorias y opcionales

Cada tabla de ID en un registro de objetos empaquetados incluirá una columna de valor de ID y puede incluir otras columnas que se definen en esta especificación como columnas opcionales o informativas (cuyo encabezado de columna no se define en esta especificación).

J.2.1 Columna de valor de ID (obligatoria)

Cada tabla de identificación en un registro de objetos empaquetados incluirá una columna de valor de ID. Los valores de ID en filas sucesivas aumentarán monótonamente. No obstante, la tabla puede terminar antes de alcanzar el número total de filas indicado por la línea de palabras clave que contiene **K-IDtamaño**. En este caso, un sistema de recepción asumirá que todos los valores de ID restantes están reservados para futuras asignaciones (como si la columna de OIDs contuviera la palabra clave "K-RFA"). Si una tabla de ID de base registrada no incluye la columna de OID opcionales que se describe a continuación, el valor de ID se utilizará como el último arco del OID.

J.2.2 Columnas de secuencias de ID y OID (opcionales)

Un registro de objetos empaquetados siempre asigna un arco OID final a cada identificador (ya sea un número asignado en la columna "OID" como se describirá a continuación, o si esa columna está ausente, el valor de ID se asigna como el arco final predeterminado). La columna OID es obligatoria en vez de opcional, si un solo valor de ID pretende representar una combinación de OID o una elección entre OID (cualquier entrada que presente una selección de OID invoca uno o más bits de ID secundarios).

Un registro de objetos empaquetados puede incluir una columna IDString, que si está presente asigna un nombre de secuencia ASCII para cada OID. Si no se proporciona ningún nombre, los sistemas deben hacer referencia al identificador por su OID (vea *J.4*). Sin embargo, muchos registros se basarán en sistemas de datos que tienen una representación ASCII para cada identificador definido, y los sistemas de recepción pueden generar opcionalmente una representación basada en esas secuencias. Si es así, la tabla de ID puede contener una columna que indique la secuencia de ID que corresponde a cada OID. Una celda de Idstring vacía significa que no hay una secuencia ASCII correspondiente asociada con el OID. Una IDstring no vacía proporcionará un nombre para cada OID invocado por la columna OID de esa fila (o un solo nombre, si no hay ninguna columna OID presente). Por lo tanto, la secuencia de operaciones de combinación y selección en una Idstring coincidirá exactamente con las de la columna OID de la fila.

Una celda de **OID** no vacía puede contener una palabra clave, una secuencia ASCII que representa (en decimal) un valor único de OID o una secuencia compuesta (en notación ABNF) que define una opción o una combinación de OID. La sintaxis detallada para secuencias OID compuestas en esta columna (que también se aplica a la columna IDstring) se define en la sección *J.3*. En lugar de contener una representación de OID simple o compuesta, una entrada de OID puede contener una de las siguientes palabras clave:

- **K-Codificacióntextual = OIDddBnn**, donde "dd" representa el penúltimo arco elegido de OID y "Bnn" indica una de las tablas de codificación de Base 10, Base 40 o Base 74. Esta entrada invoca una serie de bits de ID secundarios que sirven para dos fines:
  - Codifican un "nombre" identificador ASCII que podría no haber existido al momento en que se registró la tabla. El nombre está codificado en la sección de bits de ID secundarios como una serie de valores Base-n que representan los caracteres ASCII del nombre, precedidos por un campo de cuatro bits que indica el número de valores Base-n que siguen (se permite cero, para admitir entradas RFA como se describe a continuación).
  - El valor acumulativo de estos bits de ID secundarios, considerados como un único entero binario sin signo y convertido a decimal, es el "arco" final del OID para este identificador "de codificación textual".
- **K-Secundario = Snn**, donde "Snn" representa el ID de tabla de una tabla de ID secundaria en el mismo archivo de registro. Esto es equivalente a una entrada de OID de fila de la tabla de ID base que contiene una única lista de selección (sin otros componentes en el nivel superior), pero en lugar de enumerar estos componentes en la tabla de ID base, cada componente se enumera como una fila separada en el Tabla de ID secundaria, donde a cada uno se le puede asignar un OID, una secuencia de ID y una secuencia de formato únicos.
- **K-Propietario=OIDdd Pnn**, donde nn representa un número fijo de bits de ID secundarios que codifican un identificador de empresa opcional que indica quién escribió los datos de propiedad (una entrada de **K-Propietario = OIDddP0** indica un elemento de datos de propiedad "anónimo").
- **K-RFA = OIDddBnn**, donde "Bnn" es tal cual se definió anteriormente para la codificación textual, excepto que "B0" es una asignación válida (lo que significa que no se invocan bits de ID secundarios). Esta palabra clave representa una entrada reservada para una asignación futura, con una opción para la codificación textual del "nombre" del identificador una vez que la entidad que registró este formato de datos asigna un nombre. Los codificadores pueden utilizar esta entrada, con una longitud cero "codificación textual" de cuatro bits, hasta que se asigne un "nombre" de identificador. Se puede asignar un FormatString específico a las entradas K-RFA, o se puede utilizar la codificación a/n predeterminada.

Finalmente, cualquier entrada de OID puede terminar con un solo carácter "R" (precedido por uno o más caracteres de espacio), para indicar que un bit "Repetir" se codificará como el último bit de ID secundario invocado por la entrada. Si es "1", este bit indica que otra instancia de esta clase de identificador también está codificada (es decir, este bit actúa como si se codificara una repetición del valor de ID en una lista de ID). Si es "1", este bit es seguido por otra serie de bits de ID secundarios, para representar los detalles de esta instancia adicional del valor de ID.
Una columna de IDstring no debe contener ninguna de las entradas de palabra clave enumeradas anteriormente, y una entrada de IDstring debe estar vacía cuando la entrada de OID correspondiente contiene una palabra clave.

J.2.3 Columna FormatString (opcional)

Una tabla de ID puede definir opcionalmente las características de los datos asociados con un identificador particular, con el fin de facilitar la compactación de datos. Si está presente, la entrada FormatString especifica si un elemento de datos es totalmente numérico o alfanumérico (es decir, puede contener caracteres distintos de los dígitos decimales) y especifica una longitud fija o una longitud variable. Si no hay ninguna entrada FormatString, la característica de datos predeterminada es alfanumérica. Si no hay una entrada FormatString presente, o si la entrada no especifica una longitud, se permite cualquier longitud> = 1. A menos que se especifique una única longitud fija, la longitud de cada elemento de datos codificados se codifica en la sección de Formato auxiliar del objeto empaquetado, como se especifica en *I.7*.

Si una entrada de IDstring determinada define más de un identificador, la columna FormatString correspondiente mostrará una secuencia de formato para cada uno de esos identificadores, utilizando la misma secuencia de caracteres de puntuación (sin tener en cuenta la concatenación) que se utilizó en la IDstring correspondiente.

La secuencia de formato para un único identificador será una de los siguientes:

- Un calificador de longitud seguido de "n" (para datos siempre numéricos);
- Un calificador de longitud seguido de "an" (para datos que pueden no contener dígitos); o
- Un calificador de longitud fija, seguido de "n", seguido de uno o más caracteres de espacio, seguido de un calificador de longitud variable, seguido de "an".

Un calificador de longitud debe ser nulo (es decir, ningún calificador presente, lo que indica que cualquier longitud > = 1 es legal), un solo número decimal (que indica una longitud fija) o un rango de longitud de la forma "i * j", donde "I" representa la longitud mínima permitida del elemento de datos, "j" representa la longitud máxima permitida e i <= j. En el último caso, si se omite "j", significa que la longitud máxima es ilimitada. Los datos correspondientes a una "n" en FormatString se codifican en la subsección KLN, los datos correspondientes a una "an" en FormatString se codifican en la subsección A/N.

Cuando una instancia determinada del elemento de datos se codifica en un objeto empaquetado, su longitud se codifica en la sección de formato auxiliar como se especifica en ***I.7.2***. El valor mínimo del rango no está codificado en sí mismo, pero se especifica en la columna FormatString de la tabla de ID.

Ejemplo:

Una entrada FormatString de "3*6n" indica un elemento de datos totalmente numérico cuya longitud es siempre entre tres y seis dígitos. Una longitud determinada se codifica en dos bits, donde "00" indicaría una secuencia de dígitos cuya longitud es "3" y "11" indicaría una longitud de secuencia de seis dígitos.

J.2.4 Columna Interp (opcional)

Es posible que algunos registros deseen especificar la información necesaria para las representaciones de salida del contenido del objeto empaquetado, además de la representación OID predeterminada de los arcos de cada identificador codificado. Si esta información es invariable para una tabla en particular, el archivo de registro puede incluir líneas de palabras clave como se definió previamente. Si la interpretación varía de una fila a otra dentro de una tabla, entonces se puede agregar una columna de Interp a la tabla de ID. Esta entrada de columna, si está presente, puede contener una o más de las siguientes asignaciones de palabras clave (separadas por punto y coma), como se definieron anteriormente (vea J.1.1 y J.1.2):

- K-RootOID = urn:oid:i.j.k.l...
- K-Interpretación = secuencia
- K-ISO15434=nn

Si se utilizan, estos anulan (para un identificador particular) los valores predeterminados a nivel de archivo o los especificados en la sección Encabezado de la tabla.

J.3 Sintaxis de las columnas OID, IDstring y FormatString

En una entrada dada de la tabla de ID, las columnas OID, IDString y FormatString pueden indicar uno o más mecanismos descritos en esta sección. En J.3.1 se especifica la semántica de los mecanismos, y J.3.2 especifica la sintaxis formal para las columnas de la tabla de ID.

J.3.1 Semántica de las columnas OID, IDstring y FormatString

En las descripciones siguientes, la palabra "Identificador" significa un arco final de OID (en el contexto de la columna OID) o un nombre IDString (en el contexto de la columna IDstring). Si ambas columnas están presentes, solo la columna OID invoca realmente los bits de ID secundarios.

- Un ***componente único*** que se resuelve en un solo identificador, en cuyo caso no se invocan bits de ID secundarios adicionales.
- (Únicamente para columnas OID y IDString) Un solo componente que se resuelve en uno de una serie de identificadores estrechamente relacionados, donde la representación de la secuencia del identificador varía solo en una o más posiciones de caracteres. Esto se indica mediante el operador de ***concatenación*** "%" para introducir un rango de caracteres ASCII en una posición específica. Por ejemplo, un OID cuyo arco final se define como "391n", donde el cuarto dígito "n" puede ser cualquier dígito de "0" a "6" (incluidos los caracteres ASCII de 30hex a 36hex) está representado por el componente **391%x30 -36** (tenga en cuenta que no se permiten espacios). Una concatenación invoca el número mínimo de dígitos de ID secundarios necesarios para indicar el rango especificado. Cuando tanto una columna de OID como una columna de IDstring se llenan para una fila determinada, ambas contendrán el mismo número de concatenaciones, con los mismos rangos (de modo que los números y valores de los bits de ID secundarios invocados sean coherentes). No obstante, el valor mínimo enumerado para los dos rangos puede diferir, de modo que (por ejemplo) el dígito del OID puede variar de 0 a 3, mientras que el carácter de IDstring correspondiente puede variar de "B" a "E" si así se desea. Tenga en cuenta que el uso de la concatenación restringe inherentemente la relación entre OID e IDString, por lo que la concatenación puede no ser utilizable en todas las circunstancias (la operación de selección que se describe a continuación generalmente proporciona una alternativa).
- Una ***combinación*** de dos o más componentes identificadores en una secuencia ordenada, indicada rodeando cada componente de la secuencia entre paréntesis. Por ejemplo, una entrada de IDstring **(A)(% x30-37B) (2C)** indica que el valor de ID asociado representa una secuencia de los siguientes tres identificadores:
- Identificador "A"
- Un identificador dentro del rango "0B" a "7B" (que invoca tres bits de ID secundarios para representar la elección del carácter principal), después
- Identificador "2C

Tenga en cuenta que una combinación no invoca por sí misma ningún bit de ID secundario (a menos que uno o más de sus componentes lo hagan).

- Un componente opcional se indica rodeando el componente entre paréntesis, que puede verse como una "combinación condicional". Por ejemplo, la entrada (A) [B] [C] [D] indica que el valor de ID representa el identificador A, seguido opcionalmente por B, C o D. Una lista de opciones invoca un bit de ID secundario para cada componente en corchetes, donde un "1" indica que el componente opcional fue codificado.
- Una ***selección*** entre varios componentes incompatibles se indica separando los componentes por caracteres de diagonal. Por ejemplo, la entrada Idstring **(A/B/C/(D)(E))** indica que el valor de ID totalmente calificado representa una única opción de una lista de cuatro opciones (la cuarta de las cuales es una combinación). Una selección invoca el número mínimo de bits de ID secundarios necesarios para indicar una elección de una lista del número especificado de componentes.

En general, un OID "compuesto" o una entrada de IDstring puede contener cualquiera o todas las operaciones anteriores. No obstante, para garantizar que un único análisis de izquierda a derecha de una entrada de OID da como resultado un conjunto determinista de bits de ID secundarios (que están codificados en el mismo orden de izquierda a derecha en el que son invocados por la entrada de OID), se aplican las siguientes restricciones:

- Un identificador determinado solo puede aparecer una vez en una entrada de OID. Por ejemplo, la entrada (A) (B/A) no es válida
- Una entrada de OID puede contener como máximo una sola lista de selección
- No hay restricción en el número de combinaciones (porque no invocan bits de ID secundarios)
- No hay ninguna restricción sobre el número total de concatenaciones en una entrada de OID, pero ningún componente puede contener más de dos operadores de concatenación.
- Un componente opcional puede ser un componente de una lista de selección, pero un componente opcional puede no ser un componente compuesto y, por lo tanto, no incluirá una lista de selección ni una combinación ni concatenación.
- Una entrada de OID o IDstring no puede incluir los caracteres "(", ")", "[", "]", "%", "-" o "/", a menos que se utilice como operador como se describe anteriormente. Si uno de estos caracteres es parte de un identificador "nombre" definido del sistema de datos, entonces se representará como un solo carácter literal concatenado.

J.3.2 Sintaxis formal de las columnas OID, IDstring y FormatString

En cada entrada de la tabla de ID, el contenido de las columnas OID, IDString y FormatString se ajustará a la siguiente sintaxis para Expr, a menos que la columna esté vacía o (en el caso de la columna OID) contenga una palabra clave como se especifica en *J.2.2*. Las tres columnas comparten la misma sintaxis, excepto que la sintaxis del componente es diferente para cada columna, como se especifica a continuación. En una entrada dada de la tabla de ID, el contenido de las columnas OID, IDString y FormatString (excepto si está vacío) tendrá árboles de análisis idénticos de acuerdo con esta sintaxis, excepto que los `COMPONENTES` pueden ser diferentes. Los caracteres de espacio están permitidos (y se ignoran) en cualquier lugar de un Expr, excepto que en el interior de un `COMPONENTE` solo se permiten espacios donde se especifique explícitamente lo que se indica a continuación.

```
Expr ::= SelectionExpr | "(" SelectionExpr ")" | SelectionSubexpr

SelectionExpr ::= SelectionSubexpr ( "/" SelectionSubexpr )+

SelectionSubexpr ::= COMPONENT | ComboExpr

ComboExpr ::= ComboSubexpr+

ComboSubexpr ::= "(" COMPONENT ")" | "[" COMPONENT "]"
For the OIDs column, COMPONENT shall conform to the following grammar:
COMPONENT_OIDs ::= (COMPONENT_OIDs_Char | Concat)+

COMPONENT_OIDs_Char ::= ("0".."9")+
For the IDString column, COMPONENT shall conform to the following grammar:
COMPONENT_IDString ::= UnquotedIDString | QuotedIDString

UnquotedIDString ::= (UnQuotedIDStringChar | Concat)+

UnquotedIDStringChar ::=
  "0".."9" | "A".."Z" | "a".."z" | "_"

QuotedIDString ::= QUOTE QuotedIDStringConstituent+ QUOTE

QuotedIDStringConstituent ::=
  " " | "!" | "#".."~" | (QUOTE QUOTE)

QUOTE se refiere al carácter ASCII 34 (decimal), el carácter de comillas
dobles.
```

Cuando se utiliza la forma `QuotedIDString` para `CoMPoNENT_IDString`, los caracteres `QUOTE` inicial y final ***no*** se considerarán parte de `IDString`. Entre el principio y el final de `QUOTE`, se permiten todos los caracteres ASCII en el rango de 32 (decimal) a 126 (decimal), excepto si dos caracteres `QUOTE` en una fila denotarán un solo carácter de comillas dobles que se incluirá en el IDString.

En el formulario QuotedIDString, un carácter % no denota el operador de concatenación, sino que es solo un carácter de porcentaje incluido literalmente en IDString. Para utilizar el operador de concatenación, se debe utilizar el formulario UnquotedIDString. En ese caso, se puede utilizar un operador de concatenación degenerado (donde el carácter inicial es igual al carácter final) para incluir un carácter en el IDString que no sea uno de los caracteres enumerados para UnquotedIDStringChar.

Para la columna FormatString, el COMPONENTE se ajustará a la siguiente sintaxis:

```
COMPONENT_FormatString ::= Range? ("an" | "n")
              | FixedRange "n" " "+ VarRange "an"


Range ::= FixedRange | VarRange


FixedRange ::= Number


VarRange ::= Number "*" Number?


Number ::= ("0".."9")+
```

La sintaxis de COMPONENTE para las columnas OID e IDString hace referencia a Concat, cuya sintaxis se especifica de la siguiente manera:

```
Concat ::= "%" "x" HexChar "-" HexChar


HexChar ::= ("0"."9" | "A".."F")
```

El valor hexadecimal que sigue al guion será mayor o igual que el valor hexadecimal que precede al guion. En la columna de OID, cada valor hexadecimal debe estar en el rango de 30hex a 39hex. En la columna IDString, cada valor hexadecimal debe estar en el rango de 20hex a 7Ehex.

J.4 Representación de entrada/salida de OID

El método predeterminado para representar el contenido de un objeto empaquetado a un sistema receptor es como una serie de pares de nombre/valor, donde el nombre es un OID y el valor es la secuencia de datos decodificada asociada con ese OID. A menos que se especifique lo contrario mediante una línea de palabras clave **K-RootOID**, el OID raíz predeterminado es **urn:oid:1.0.15961.ff,** donde **ff** es el formato de datos codificado en el DSFID. El arco final del OID es (de forma predeterminada) el valor de ID, pero esto suele anularse por una entrada en la columna de OID. Observe que un indicador de aplicación codificado (vea *I.5.3.1*) puede cambiar **ff** del valor indicado por el DSFID.

Si está respaldado por información en la columna IDstring de la tabla de ID, un sistema de recepción puede traducir la salida de OID a varios formatos alternativos, basándose en la representación de IDString de los OID. Uno de estos formatos, como se describe en ISO/IEC 15434, requiere como información adicional un identificador de formato de dos dígitos; un registro de tabla puede proporcionar esta información utilizando la palabra clave **K-ISO15434** como se describe anteriormente.

La combinación de la palabra clave K-RootOID y la columna OID proporciona a la entidad registradora la capacidad de asignar OID a los identificadores del sistema de datos sin tener en cuenta cómo están realmente codificados y, por lo tanto, se puede aplicar la misma asignación de OID independientemente del método de acceso.

J.4.1 Representación de salida de "OID de valor de ID"

Si el sistema receptor no tiene acceso a la tabla de ID relevante (posiblemente porque se ha registrado recientemente), el decodificador de objetos empaquetados no tendrá suficiente información para convertir el valor de ID (más los bits de ID secundarios) al OID deseado. Para facilitar la introducción de tablas nuevas o externas, los codificadores tienen la opción de seguir las reglas de "uso restringido" (vea *I.5.3.2*).

Cuando un sistema receptor ha decodificado un objeto empaquetado codificado siguiendo las reglas de "uso restringido", pero no tiene acceso a la tabla de ID indicada, construirá un "OID de valor de ID" con el siguiente formato:

urn:oid:1.0.15961.300.ff.bb.idval.secbits

donde **1.0.15961.300** es un OID raíz con un formato de datos reservado de "300" que nunca se codifica en un DSFID, pero se utiliza para distinguir un "OID de valor de ID" de un OID verdadero (como se habría utilizado si la tabla de ID estuviese disponible). El valor reservado de 300 después tiene el formato de datos de la tabla codificada (**ff**) (que puede ser diferente del predeterminado del DSFID), la ID de la tabla (**bb**) (siempre "0", a menos que se indique lo contrario mediante un indicador de aplicación codificado), el valor de ID codificado y la representación decimal de los bits de ID secundarios invocados. Este proceso crea un OID único para cada valor de ID único totalmente calificado. Por ejemplo, utilizando la tabla de ID hipotética que se muestra en el Anexo *L* (pero asumiendo, con fines ilustrativos, que el OID raíz especificado en la tabla es **urn:oid:1.0.12345.9**, entonces una ID de "CANTIDAD" con un cuarto dígito de "2" tiene un verdadero OID de:

urn:oid:1.0.12345.9.3912

y un "OID de valor de ID" de

urn:oid:1.0.15961.300.9.0.51.2

Cuando un único valor de ID representa identificadores de múltiples componentes a través de combinaciones o componentes opcionales, sus múltiples OID y secuencias de datos se representarán por separado, cada uno usando el mismo "OID de valor de ID" (hasta el arco de bits de ID secundario), pero agregando como un arco final el número de componente (comenzando con "1" para el primer componente decodificado bajo ese valor de ID).

Si el sistema de decodificación encuentra un objeto empaquetado que hace referencia a una tabla de ID que no está disponible para el decodificador, pero el codificador eligió no establecer el bit de "uso restringido" en el indicador de aplicación, el decodificador descartará el objeto empaquetado o retransmitirá todo el objeto empaquetado al sistema receptor como una única entidad binaria no codificada, una secuencia de octetos de la longitud especificada en el campo ObjectLength del objeto empaquetado. El OID para un objeto empaquetado no codificado será **urn:oid:1.0.15961.301.ff.n**, donde "301" es un formato de datos reservado para indicar un objeto empaquetado no codificado, "ff" será el formato de datos codificado en el DSFID en el inicio de la memoria, y un arco final "n" opcional puede incrementarse secuencialmente para distinguir entre múltiples objetos empaquetados no codificados en la misma memoria portadora de datos.

K Tablas de codificación de objetos empaquetados

Los objetos empaquetados utilizan principalmente dos bases de codificación:

- Base 10, que codifica cada uno de los dígitos del “0” al “9” en un valor de Base 10
- Base 30, que codifica las letras mayúsculas y la puntuación seleccionable en un valor de Base 30, y codifica los caracteres de puntuación y control del resto del conjunto de caracteres ASCII en dos valores de base 30 (utilizando un mecanismo de cambio)

Para situaciones en las que un alto porcentaje de los caracteres no numéricos de los datos de entrada requeriría pares de valores de base 30, también se definen dos bases alternativas, Base 74 y Base 256:

- Los valores del conjunto de Base 74 corresponden al subconjunto invariante de la norma ISO 646 (que incluye el conjunto de caracteres GS1), pero con los dígitos eliminados y con la adición de GS y <espacio> (GS se admite para usos que no sean como delimitador de datos).
- Los valores en el conjunto de Base 256 pueden transmitir octetos sin interpretación de caracteres gráficos, o “valores ASCII extendidos” como se define en la norma ISO 8859-6 o UTF-8 (la interpretación puede establecerse en la Tabla de ID registrada para una aplicación) Se admiten los caracteres “0” a “9” (valores ASCII 48 a 57) y, por lo tanto, un codificador puede codificar los dígitos utilizando un prefijo o sufijo (en Base 256) o utilizando un mapa de caracteres (en Base 10) . Tenga en cuenta que en los datos GS1, FNC1 está representado por ASCII <GS> (valor de octeto 29dec).

Por último, hay situaciones en las que la eficacia de la compactación se puede mejorar mediante la codificación de la longitud de ejecución de los indicadores de base, en vez de mediante los bits del mapa de caracteres, cuando una larga serie de caracteres se puede clasificar en una sola base. Para facilitar esa clasificación, se agregan bases de "extensión" adicionales, solo para su uso en Ejecuciones de prefijos y sufijos.

- Para admitir la codificación de longitud de ejecución de una secuencia principalmente numérica con algunas letras intercaladas, se define una Base 13, según la Tabla B-2
- Dos de estas bases de extensión (Base 40 y Base 84) se definen simplemente, en el sentido de que extienden las correspondientes bases no numéricas (Base 30 y Base 74, respectivamente) para incluir también los diez dígitos decimales. Las entradas adicionales, para los caracteres “0” a “9”, se agregan como los siguientes diez valores secuenciales (valores 30 a 39 para Base 40 y valores 74 a 83 para Base 84).
- La versión "extendida" de Base 256 se define como Base 40. Esto permite a un codificador la opción de codificar algunos caracteres de control ASCII o caracteres ASCII superiores en Base 256, mientras usa un prefijo o sufijo para codificar de manera más eficiente los caracteres no numéricos restantes.

El número de bits necesarios para codificar varios números de caracteres Base 10, Base 16, Base 30, Base 40, Base 74 y Base 84 se muestra en la Figura B-1. En todos los casos, se establece un límite en el tamaño de un solo grupo de entrada, seleccionado para generar un grupo de no más de 20 octetos.

Figura K-1 Número requerido de bits para un número dado de valores base “N”

```
/* Base10 encoding accepts up to 48 input values per group: */
static const unsigned char bitsForNumBase10[] = {
/* 0 - 9 */  0,  4,  7, 10, 14, 17, 20, 24, 27, 30,
/* 10 - 19 */  34, 37, 40, 44, 47, 50, 54, 57, 60, 64,
/* 20 - 29 */  67, 70, 74, 77, 80, 84, 87, 90, 94, 97,
/* 30 - 39 */ 100, 103, 107, 110, 113, 117, 120, 123, 127, 130,
/* 40 - 48 */ 133, 137, 140, 143, 147, 150, 153, 157, 160};

/* Base13 encoding accepts up to 43 input values per group: */
static const unsigned char bitsForNumBase13[] = {
/* 0 - 9 */  0,  4,  8, 12, 15, 19, 23, 26, 30, 34,
/* 10 - 19 */  38, 41, 45, 49, 52, 56, 60, 63, 67, 71,
/* 20 - 29 */  75, 78, 82, 86, 89, 93, 97, 100, 104, 108,
/* 30 - 39 */ 112, 115, 119, 123, 126, 130, 134, 137, 141, 145,
/* 40 - 43 */ 149, 152, 156, 160 };

/* Base30 encoding accepts up to 32 input values per group: */
static const unsigned char bitsForNumBase30[] = {
/* 0 - 9 */  0,  5, 10, 15, 20, 25, 30, 35, 40, 45,
/* 10 - 19 */  50, 54, 59, 64, 69, 74, 79, 84, 89, 94,
/* 20 - 29 */  99, 104, 108, 113, 118, 123, 128, 133, 138, 143,
/* 30 - 32 */ 148, 153, 158};

/* Base40 encoding accepts up to 30 input values per group: */
static const unsigned char bitsForNumBase40[] = {
/* 0 - 9 */  0,  6, 11, 16, 22, 27, 32, 38, 43, 48,
/* 10 - 19 */  54, 59, 64, 70, 75, 80, 86, 91, 96, 102,
/* 20 - 29 */ 107, 112, 118, 123, 128, 134, 139, 144, 150, 155,
/* 30 */ 160 };

/* Base74 encoding accepts up to 25 input values per group: */
static const unsigned char bitsForNumBase74[] = {
/* 0 - 9 */  0,  7, 13, 19, 25, 32, 38, 44, 50, 56,
/* 10 - 19 */  63, 69, 75, 81, 87, 94, 100, 106, 112, 118,
/* 20 - 25 */ 125, 131, 137, 143, 150, 156 };

/* Base84 encoding accepts up to 25 input values per group: */
static const unsigned char bitsForNumBase84[] = {
/* 0 - 9 */  0,  7, 13, 20, 26, 32, 39, 45, 52, 58,
/* 10 - 19 */  64, 71, 77, 84, 90, 96, 103, 109, 116, 122,
/* 20 - 25 */ 128, 135, 141, 148, 154, 160 };
```

Tabla K-1 Conjunto de caracteres de base 30

Val	Conjunto básico		Conjunto de cambio 1		Conjunto de cambio 2	
	Car	Decimal	Car	Decimal	Car	Decimal
0	A-Punc[1]	N/A	NUL	0	espacio	32

Val	Conjunto básico		Conjunto de cambio 1		Conjunto de cambio 2	
1	A	65	SOH	1	!	33
2	B	66	STX	2	“	34
3	C	67	ETX	3	#	35
4	D	68	EOT	4	$	36
5	E	69	ENQ	5	%	37
6	F	70	ACK	6	&	38
7	G	71	BEL	7	`	39
8	H	72	BS	8	(	40
9	I	73	HT	9	)	41
10	J	74	LF	10	*	42
11	K	75	VT	11	+	43
12	L	76	FF	12	,	44
13	M	77	RC	13	-	45
14	N	78	SO	14	.	46
15	O	79	SI	15	/	47
16	P	80	DLE	16	:	58
17	Q	81	ETB	23	;	59
18	R	82	ESC	27	<	60
19	S	83	FS	28	=	61
20	T	84	GS	29	>	62
21	U	85	RS	30	?	63
22	V	86	US	31	@	64
23	W	87	no válido	N/A	\	92
24	X	88	no válido	N/A	^	94
25	Y	89	no válido	N/A	_	95
26	Z	90	[	91	`	96
27	Cambio 1	N/A	]	93	|	124
28	Cambio 2	N/A	{	123	~	126
29	P-Punc[2]	N/A	}	125	no válido	N/A

Nota 1: El carácter de **puntuación especificado por la aplicación** (valor 0 del conjunto básico) se define de forma predeterminada como el carácter de guion ASCII (45dec), pero puede redefinirse mediante un formato de datos registrado.

Nota 2: Carácter de **puntuación programable** (valor 29 del conjunto básico): la primera aparición de P-Punc en los datos alfanuméricos de un objeto empaquetado, ya sea que la primera aparición esté compactada en el segmento de Base 30 o en el segmento de Base 40, actúa como un <Cambio 2>, y también “programa” el carácter para que sea representado por la segunda y subsiguientes apariciones de P-Punc (en cualquier segmento) para el resto de los datos alfanuméricos en ese objeto empaquetado. El valor de Base 30 o Base 40 inmediatamente después de esa primera aparición se interpreta utilizando la columna Cambio 2 (Puntuación) y se asigna a las instancias posteriores de P-Punc para el objeto empaquetado.

Tabla K-2 Conjunto de caracteres de base 13

Valor	Conjunto básico		Conjunto de cambio 1		Conjunto de cambio 2		Conjunto de cambio 3	
	Car	Decimal	Car	Decimal	Car	Decimal	Car	Decimal
0	0	48	A	65	N	78	espacio	32
1	1	49	B	66	O	79	$	36
2	2	50	C	67	P	80	%	37
3	3	51	D	68	Q	81	&	38
4	4	52	E	69	R	82	*	42
5	5	53	F	70	S	83	+	43
6	6	54	G	71	T	84	,	44
7	7	55	H	72	U	85	-	45
8	8	56	I	73	V	86		46
9	9	57	J	74	W	87	/	47
10	Shift1	N/A	K	75	X	88	?	63
11	Shift2	N/A	L	76	Y	89	_	95
12	Shift3	N/A	M	77	Z	90	<GS>	29

Tabla K-3 Conjunto de caracteres de base 40

Val	Conjunto básico		Conjunto de cambio 1		Conjunto de cambio 2	
	Car	Decimal	Car	Decimal	Car	Decimal
0	Vea Tabla K-1					
…	…					
29	Vea Tabla K-1					
30	0	48				
31	1	49				
32	2	50				
33	3	51				
34	4	52				
35	5	53				
36	6	54				
37	7	55				
38	8	56				
39	9	57				

Tabla K-4 Conjunto de caracteres

Val	Car	Decimal	Val	Car	Decimal	Val	Car	Decimal
0	GS	29	25	F	70	50	d	100
1	!	33	26	G	71	51	e	101
2	"	34	27	H	72	52	f	102
3	%	37	28	I	73	53	g	103
4	&	38	29	J	74	54	h	104
5	'	39	30	K	75	55	i	105

Val	Car	Decimal	Val	Car	Decimal	Val	Car	Decimal
6	(	40	31	L	76	56	j	106
7	)	41	32	M	77	57	k	107
8	*	42	33	N	78	58	l	108
9	+	43	34	O	79	59	m	109
10	,	44	35	P	80	60	n	110
11	-	45	36	Q	81	61	o	111
12		46	37	R	82	62	p	112
13	/	47	38	S	83	63	q	113
14		58	39	T	84	64	r	114
15	;	59	40	U	85	65	s	115
16	<	60	41	V	86	66	t	116
17	=	61	42	W	87	67	u	117
18	>	62	43	X	88	68	v	118
19	?	63	44	Y	89	69	w	119
20	A	65	45	Z	90	70	x	120
21	B	66	46	_	95	71	y	121
22	C	67	47	a	97	72	z	122
23	D	68	48	b	98	73	espacio	32
24	E	69	49	c	99			

Tabla K-5 Conjunto de caracteres de base 84

Val	Car	Decimal	Val	Car	Decimal	Val	Car	Decimal
0	FNC1	N/A	25	F		50	d	
1-73	Vea Tabla K-4							
74	0	48	78	4	52	82	8	56
75	1	49	79	5	53	83	9	57
76	2	50	80	6	54			
77	3	51	81	7	55			

L Codificación de objetos empaquetados (no normativa)

Para ilustrar algunas de las técnicas que se pueden invocar al codificar un objeto empaquetado, los siguientes datos de entrada de muestra consisten en elementos de datos de un sistema de datos hipotético. Estos datos representan:

- Una fecha de vencimiento (OID 7) del 31 de octubre de 2006, representada como un número de seis dígitos 061031.
- Una cantidad por pagar (OID 3n) de 1234,56 euros, representada como una secuencia de dígitos 978123456 ("978" es el código de país ISO que indica que la cantidad a pagar está en euros). Como se muestra en la Tabla L-1, este elemento de datos es totalmente numérico, con al menos 4 dígitos y un máximo de 18 dígitos. En este ejemplo, el OID "3n" será "32", donde el "2" en el nombre del elemento de datos indica que el punto decimal está ubicado dos dígitos a la derecha.
- Un número de lote (OID 1) de 1A23B456CD

La aplicación presentará la entrada anterior al codificador como una lista de pares de valor/OID. Los datos de entrada resultantes, representados a continuación como una sola secuencia de datos (donde cada arco final de OID se muestra entre paréntesis) son:

(7)061031(32)978123456(1)1A23B456CD

El ejemplo utiliza una tabla de ID hipotética. En esta tabla hipotética, cada valor de ID es un índice de siete bits en la tabla de ID de base; las entradas relevantes para este ejemplo se muestran en la Tabla L-1.

La codificación se realiza en los siguientes pasos:

- Se codificarán tres elementos de datos, utilizando la Tabla L-1.
- Como se muestra en la columna IDstring de la tabla, la combinación de OID 7 y OID 1 se admite de manera eficiente (porque se ve comúnmente en las aplicaciones) y, por lo tanto, el codificador reordena la entrada para que 7 y 1 sean adyacentes y en el orden indicado en la columna OID:
- (7)061031(1)1A23B456CD(32)978123456
- Ahora, a este par de OID se le puede asignar un valor de ID único de 125 (decimal). La columna FormatString para esta entrada muestra que los datos codificados siempre constarán de una secuencia de 6 dígitos de longitud fija, seguida de una secuencia alfanumérica de longitud variable.
- También como se muestra en la Tabla L-1, OID 3n tiene un valor de ID de 51 (decimal). La columna OID para esta entrada muestra que el OID se forma concatenando "3" con un sufijo que consta de un solo carácter en el rango de 30hex a 39hex (es decir, un dígito decimal). Dado que ese es un rango de diez posibilidades, será necesario codificar un número de cuatro bits en la sección de ID secundaria para indicar qué carácter de sufijo se eligió. La columna FormatString para esta entrada muestra que sus datos son numéricos de longitud variable; la información de longitud variable requerirá la codificación de cuatro bits en la sección de formato auxiliar.
- Dado que solo se utiliza un pequeño porcentaje de la tabla de ID de 128 entradas en este objeto empaquetado, el codificador elige un formato de lista de ID, en lugar de un formato de mapa de ID. Como este es el formato predeterminado, no se requiere la sección de Indicadores de formato.
- Estos resultados se muestran en la sección de Información del objeto:
  - EBV-6 (ObjectLength): el valor es TBD en esta etapa del proceso de codificación
  - Bit indicador de relleno: TBD en esta etapa
  - EBV-3 (numberOfIDs) de 001 (lo que significa que seguirán dos valores de ID)
  - Una lista de ID, que incluye:
    - Primer valor de ID: 125 (dec) en 7 bits, que representan OID 7 seguido de OID 1
    - Segundo valor de ID: 51 (decimal) en 7 bits, que representa OID 3n
- Una sección de ID secundaria se codifica como "0010", lo que indica el "2" final del 3n OID. Este "2" significa que dos dígitos siguen al punto decimal implícito, pero esa información no es necesaria para codificar o decodificar el objeto empaquetado.
- A continuación, se codifica una sección de formato auxiliar. Se codifica un bit "1" inicial, que invoca el método de compactación de objetos empaquetados.

- De los tres OID, solo OID (3n) requiere información de formato auxiliar codificada: un patrón de cuatro bits de "0101" (que representa "seis" dígitos de longitud variable, ya que "uno" es la primera opción permitida, un patrón de "0101" denota "seis").
- A continuación, el codificador codifica el primer elemento de datos, para el OID 7, que se define como un elemento de datos de seis dígitos de longitud fija. Los seis dígitos de la secuencia de datos de origen son "061031", que se convierten en una secuencia de seis valores de base 10 restando 30hex de cada carácter de la secuencia (los valores resultantes se indican como valores v5 a v0 en la fórmula siguiente). Posteriormente, estos se convierten en un solo valor binario, utilizando la siguiente fórmula:
  - $10^5 * v_5 + 10^4 * v_4 + 10^3 * v_3 + 10^2 * v_2 + 10^1 * v_1 + 10^0 * v_0$

    De acuerdo con la Figura K-1, un número de seis dígitos siempre se codifica en 20 bits (independientemente de los ceros iniciales en la entrada), lo que da como resultado una secuencia binaria de:
    "0000 11101110 01100111"
- El siguiente elemento de datos es para OID 1, pero dado que la tabla indica que los datos de este OID son alfanuméricos, la codificación en el objeto empaquetado se aplaza hasta que se codifiquen todos los datos numéricos de longitud conocida.
- A continuación, el codificador encuentra que OID 3n está definido por la Tabla L-1 como todo numérico, cuya longitud de 9 (en este ejemplo) se codificó como (9 - 4 = 5) en cuatro bits dentro de la subsección de formato auxiliar. Por lo tanto, se codifica una subsección numérica de longitud conocida para este elemento de datos, que consta de un patrón de bits de valor binario que codifica 9 dígitos. Con la Figura K-1 en el Anexo *K*, el codificador determina que es necesario codificar 30 bits para representar un número de 9 dígitos como valor binario. En este ejemplo, el valor binario equivalente a "978123456" es la secuencia binaria de 30 bits:
  "11101001001100111110101100000"
- En este punto, la codificación de la subsección numérica de longitud conocida de la Sección de datos está completa.

Considere que, hasta ahora, el número total de bits codificados es (3 + 6 + 1 + 7 + 7 + 4 + 5 + 20 + 30) o 83 bits, lo que representa la sección de longitud IDLPO (suponiendo que un solo vector EBV-6 sigue siendo suficiente para codificar la longitud del objeto empaquetado), dos valores de ID de 7 bits, las secciones de identificación secundaria y formato auxiliar, y dos campos binarios compactados numéricos de longitud conocida.

En esta etapa, solo queda por codificar una secuencia de datos no numéricos (para OID 1) en la subsección alfanumérica. La secuencia de datos de origen de 10 caracteres es "1A23B456CD". Esta secuencia no contiene caracteres que requieran un cambio de base 30 fuera del conjunto de caracteres básico de base 30, por lo que se selecciona base 30 en la base no numérica (por lo que el primer bit de la subsección alfanumérica se establece en "0", respectivamente). La secuencia de datos no tiene subsecuencias con seis o más caracteres sucesivos de la misma base, por lo que los dos bits siguientes se establecen en "00" (lo que indica que ni un prefijo ni un sufijo están codificados en longitud de ejecución). Por lo tanto, se debe codificar un mapa de caracteres completo de 10 bits. Su patrón de bits específico es "0100100011", que indica la secuencia específica de dígitos y no dígitos en la secuencia de datos de origen "1A23B456CD".

Hasta este punto, la subsección alfanumérica contiene la secuencia de 13 bits "0 00 0100100011". A partir del Anexo *K*, se puede determinar que las longitudes de las dos secuencias de bits finales (que codifican los componentes de base 10 y base 30 de la secuencia de datos de origen) son 20 bits (para los seis dígitos) y 20 bits (para las cuatro letras en mayúsculas usando la base 30). Los seis dígitos de la secuencia de datos de origen "1A23B456CD" son "123456", que se codifica en una secuencia de 20 bits de:
"00011110001001000000"

que se adjunta al final de la secuencia de 13 bits citada al comienzo de este párrafo.

Los cuatro no dígitos de la secuencia de datos de origen son "ABCD", que se convierten (utilizando la Tabla K-1) en una secuencia de cuatro valores de base 30 1, 2, 3 y 4 (indicados como valores v3 a v0 en la siguiente fórmula. Posteriormente, estos se convierten en un solo valor binario, utilizando la siguiente fórmula:

$30^3 * v_3 + 30^2 * v_2 + 30^1 * v_1 + 30^0 * v_0$

En este ejemplo, la fórmula se calcula como (27000 * 1 + 900 * 2 + 30 * 3 + 1 * 4) que es igual a 070DE (hexadecimal) codificado como la secuencia de 20 bits "00000111000011011110" que se adjunta al final de la secuencia anterior de 20 bits. Por lo tanto, la sección alfanumérica contiene un total de (13 + 20 + 20) o 53 bits, agregados inmediatamente después de los 83 bits anteriores, para un total general de 136 bits significativos en el objeto empaquetado.

El paso de codificación final es calcular la longitud completa del objeto empaquetado (para codificar el EBV-6 dentro de la sección de longitud) y rellenar el último byte (si es necesario). El dividir 136 entre ocho muestra que se requiere un total de 17 bytes para contener el objeto empaquetado y que no se requieren bits de relleno en el último byte. Por lo tanto, la parte de EBV-6 de la sección de longitud es "010001", donde este valor de EBV-6 indica 17 bytes en el objeto. Después de eso, el bit indicador de relleno se establece en "0", lo que indica que no hay bits de relleno presentes en el último byte de datos.

El proceso de codificación completo se puede resumir de la siguiente manera:

Entrada original: (7)061031(32)978123456(1)1A23B456CD

Reordenado como: (7)061031(1)1A23B456CD(32)978123456

SECCIÓN DE INDICADORES DE FORMATO: (vacío)

SECCIÓN DE INFORMACIÓN DEL OBJETO:

ebvObjectLen: 010001

paddingPresent: 0

ebvNumIDs: 001

IDvals: 1111101 0110011

SECCIÓN DE ID SECUNDARIA:

IDbits: 0010

SECCIÓN DE FORMATO AUX:

auxFormatbits: 1 0101

SECCIÓN DE DATOS:

KLnumeric: 0000 11101110 01100111 11101001100 11111010 11000000

ANheader: 0

ANprefix: 0

ANsuffix: 0

ANmap: 01 00100011

ANdigitVal: 01000000 11100010 0001

ANnonDigitsVal: 11011110 01110000 0000

Relleno: ninguno

Total de bits en el objeto empaquetado: 136; cuando los bytes están alineados: 136

Salida como: 44 7E B3 2A 87 73 3F 49 9F 58 01 23 1E 24 00 70 DE

La Tabla L-1 muestra el subconjunto relevante de una tabla de ID hipotética para un formato de datos 99 hipotético registrado por ISO.

Tabla L-1 Tabla de ID base hipotética, para el ejemplo del Anexo L

K-Versión = 1.0			
K-TableID = F99B0			
K-RootOID = urn:oid:1.0.15961.99			
K-IDsize = 128			
IDvalue	OIDs	Título de datos	FormatString
3	1	LOTE	1*20an

K-Versión = 1.0			
8	7	EXPIRA O CADUCA	6n
51	3%x30-39	CANTIDAD	4*18n
125	(7) (1)	VENCIMIENTO + LOTE	(6n) (1*20an)
K-TableEnd = F99B0			

M Decodificación de objetos empaquetados (no normativa)

M.1 Descripción general

El proceso de decodificación comienza decodificando el primer byte de la memoria como un DSFID. Si los dos bits iniciales indican el método de acceso a los objetos empaquetados, se aplica el resto de este Anexo. A partir del resto del octeto u octetos DSFID, determine el formato de datos, que se aplicará como formato de datos predeterminado para todos los objetos empaquetados en esta memoria. A partir del formato de datos, determine la tabla de ID predeterminada que se utilizará para procesar los valores de ID en cada objeto empaquetado.

Por lo general, el decodificador realiza un primer pase a través de la lista de valores de ID inicial, como se describió anteriormente, para completar la lista de identificadores. Si el decodificador encuentra algún identificador de interés en un objeto empaquetado (o si se le ha pedido que informe todas las secuencias de datos de la memoria de una etiqueta), entonces deberá registrar las longitudes fijas implícitas (de la tabla de ID) y las longitudes variables codificadas (de la subsección Formato auxiliar), para analizar los datos comprimidos del objeto empaquetado. El decodificador, al registrar cualquier patrón de bits de longitud variable, primero debe convertirlos a longitudes de secuencia variables según la tabla (por ejemplo, un patrón de tres bits puede indicar una longitud de secuencia variable en el rango de dos a nueve).

Comenzando en la primera posición alineada por bytes después del final del DSFID, analice el contenido de la memoria restante hasta el final de los datos codificados, repitiendo el resto de esta sección hasta que se alcance un patrón de terminación.

Determine a partir del patrón de bits principal (vea *I.4*) cuál de las siguientes condiciones se aplica:

1. no hay más objetos empaquetados en la memoria (si el patrón de 8 bits principal está en ceros, esto indica el patrón de terminación)
2. están presentes uno o más bytes de relleno Si hay relleno, omita los bytes de relleno, que se describen en el Anexo *I*, y examine el primer byte que no es de relleno.
3. se codifica un puntero de directorio. Si está presente, registre el desplazamiento indicado por los siguientes bytes y luego continúe examinando desde el siguiente byte en la memoria.
4. está presente una sección de indicadores de formato; en este caso, procese esta sección de acuerdo con el formato descrito en el Anexo *I*
5. un objeto empaquetado con formato predeterminado comienza en esta localización

Si el objeto empaquetado tenía una sección de indicadores de formato, entonces esta sección puede indicar que el objeto empaquetado tiene el formato de mapa de ID; de lo contrario, tiene el formato de lista de ID. De acuerdo con el formato indicado, analice la sección Información del objeto para determinar la longitud del objeto y la información de ID contenida en el objeto empaquetado. Vea el Anexo *I* para conocer los detalles de los dos formatos. Independientemente del formato, este paso da como resultado una longitud de objeto conocida (en bits) y una lista ordenada de los valores de identificación codificados en el objeto empaquetado. A partir de la tabla de ID que rige, determine la lista de características para cada ID (como la presencia y el número de bits de ID secundarios).

Analice la sección de ID secundaria del objeto, con base en el número de bits de ID secundaria invocados por cada valor de ID en secuencia. A partir de esta información, cree una lista de los valores de identificación completos (FQIDV) que están codificados en el objeto empaquetado.

Analice la sección de formato auxiliar del objeto, con base en el número de bits de formato auxiliar invocados por cada FQIDV en secuencia.

Analizar la sección de datos del objeto empaquetado:

1. Si uno o más de los FQIDV indican datos totalmente numéricos, la sección de datos del objeto empaquetado contiene una subsección numérica de longitud conocida, en la que las secuencias de dígitos de estos elementos totalmente numéricos se han codificado como una serie de cantidades binarias. Utilizando la longitud conocida de cada uno de estos elementos de datos totalmente numéricos, analice el número correcto de bits para cada elemento de datos y convierta cada conjunto de bits en una secuencia de dígitos decimales.
2. Si (después de analizar las secciones anteriores) uno o más de los FQIDV indican datos alfanuméricos, entonces la sección de datos del objeto empaquetado contiene una subsección alfanumérica, en la que las secuencias de caracteres de estos elementos alfanuméricos se han concatenado y codificado en la estructura definida en el Anexo *I*. Decodifique estos datos utilizando el procedimiento "Decodificación de datos alfanuméricos" que se describe a continuación.

3. Para cada FQIDV en la secuencia decodificada
4. convierta el FQIDV en un OID, agregando la secuencia OID definida en la Tabla de ID del formato registrado a la secuencia OID raíz definida en esa Tabla de ID (o al OID raíz predeterminado, si no hay ninguno definido en la tabla)
5. Complete el par OID/Valor analizando la siguiente secuencia de caracteres decodificados La longitud de esta secuencia se determina directamente de la tabla de ID (si el FQIDV se especifica como longitud fija) o de una entrada correspondiente codificada dentro de la sección de formato auxiliar.

M.2 Decodificación de datos alfanuméricos

Dentro de la subsección alfanumérica de un objeto empaquetado, el número total de caracteres de datos no está codificado, ni la longitud de bits del mapa de caracteres, ni las longitudes de bits de las secciones binarias siguientes (que representan los valores binarios numéricos y no numéricos) Como resultado, el decodificador debe seguir un procedimiento específico para analizar correctamente la sección alfanumérica.

Al decodificar la subsección A/N utilizando este procedimiento, el decodificador primero contará el número de valores sin mapas de bits en cada base (como lo indican las diversas ejecuciones de prefijo y sufijo), y (a partir de ese recuento) determinará el número de bits necesario para codificar estos números de valores en estas bases. A continuación, el procedimiento puede calcular, a partir del número restante de bits, el número de bits del mapa de caracteres codificados explícitamente. Después de decodificar por separado los diversos campos binarios (un campo para cada base que se utilizó), el decodificador "reintercala" los caracteres ASCII decodificados en el orden correcto.

El procedimiento de decodificación de la subsección A/N es el siguiente:

- Determine el número total de bits que no son de relleno en el objeto empaquetado, como se describe en la sección *I.8.2*
- Mantenga un conteo del número total de bits analizados hasta el momento, ya que se procesa cada una de las subsecciones antes de la subsección alfanumérica
- Analice los bits del encabezado inicial de la subsección alfanumérica, hasta el mapa de caracteres, pero sin incluirlo, y agregue este número al valor anterior de TotalBitsParsed.
- Inicie un DigitsCount al número total de valores de base 10 indicados por el prefijo y el sufijo (que puede ser cero)
- Inicie un ExtDigitsCount al número total de valores de base 13 indicados por el prefijo y el sufijo (que puede ser cero)
- Inicie un NonDigitsCount al número total de valores de base 30, base 74 o base 256 indicados por el prefijo y el sufijo (que puede ser cero)
- Inicie un ExtNonDigitsCount al número total de valores base 40 o base 84 indicados por el prefijo y el sufijo (que puede ser cero)
- Calcular conteos de bits de base extendida: Mediante el uso de las tablas en el Anexo *K*, calcule dos números:
  - ExtDigitBits, el número de bits necesarios para codificar el número de valores de base 13 indicados por ExtDigitsCount, y
  - ExtNonDigitBits, el número de bits necesarios para codificar el número de valores de base 40 (o base 84) indicados por ExtNonDigitsCount
  - Agregue ExtDigitBits y ExtNonDigitBits a TotalBitsParsed
- Cree una secuencia de bits PrefixCharacterMap, una secuencia de cero o más pares de mapa de caracteres de base cuádruple, como lo indican los bits de Prefijo recién analizados. Utilice pares de bits de base cuádruple definidos de la siguiente manera:
  - "00" indica un valor de base 10;
  - "01" indica un carácter codificado en Base 13;
  - "10" indica la base no numérica que se seleccionó anteriormente en el encabezado A/N, y
  - "11" indica la versión extendida de la base no numérica que se seleccionó anteriormente
- Cree una secuencia de bits SuffixCharacterMap, una secuencia de cero o más pares de mapas de caracteres de base cuádruple, como lo indican los bits de Sufijo recién analizados.

- Inicie la secuencia de bits FinalCharacterMap y la secuencia de bits MainCharacterMap en una secuencia vacía
- **Calcular conteos de bits en ejecución:** Mediante el uso de las tablas en el Anexo *B*, calcule dos números:
  - DigitBits, el número de bits necesarios para codificar el número de valores de base 10 indicados actualmente por DigitsCount, y
  - NonDigitBits, el número de bits necesarios para codificar el número de valores de base-30 (o base 74 o base-256) indicados actualmente por NonDigitsCount
- establezca AlnumBits igual a la suma de DigitBits más NonDigitBits
- si la suma de TotalBitsParsed y AlnumBits es igual al número total de bits que no son de relleno en el objeto empaquetado, entonces no quedan más bits por analizar del mapa de caracteres y, por lo tanto, los patrones de bits restantes, que representan valores binarios, están listos para convertirse a los valores base extendidos o los valores de base 10, base 30, base 74, base-256 (siga los pasos de **Decodificación final** a continuación). De lo contrario, obtenga el siguiente bit codificado del mapa de caracteres codificado, convierta el bit en un par de bits de base cuádruple convirtiendo cada "0" en "00" y cada "1" en "10", agregue el par al final de la secuencia de bits MainCharacterMap, y:
  - Si el bit de mapa codificado era "0", incremente el DigitsCount,
  - De lo contrario, si es "1", aumentar NonDigitsCount
  - Vuelva al paso anterior **Calcular conteos de bits en ejecución** y continúe
- **Pasos finales de decodificación:** una vez que los bits del mapa de caracteres codificados se hayan analizado completamente:
  - Obtenga el siguiente conjunto de cero o más bits, cuya longitud está indicada por ExtDigitBits. Convierta este número de bits de valores binarios a una serie de valores de base 13 y almacene la matriz de valores resultante como ExtDigitVals.
  - Obtenga el siguiente conjunto de cero o más bits, cuya longitud está indicada por ExtNonDigitBits. Convierta este número de bits de valores binarios a una serie de valores de base 40 o base 84 (dependiendo de la selección indicada en el encabezado A/N) y almacene la matriz de valores resultante como ExtNonDigitVals.
  - Obtenga el siguiente conjunto de bits, cuya longitud está indicada por DigitBits. Convierta este número de bits de valores binarios a una serie de valores de base 10 y almacene la matriz de valores resultante como DigitVals.
  - Obtenga el siguiente conjunto de bits, cuya longitud está indicada por NonDigitBits. Convierta este número de bits de valores binarios a una serie de valores de base 30 o base 74 o base 256 (dependiendo del valor de los primeros bits de la subsección alfanumérica) y almacene la matriz de valores resultante como NonDigitVals.
  - Cree la secuencia de bits FinalCharacterMap copiando en ella, en este orden, la secuencia de bits PrefixCharacterMap previamente creada, luego la secuencia MainCharacterMap y finalmente agregue la secuencia de bits SuffixCharacterMap creada previamente al final de la secuencia FinalCharacterMap.
  - Cree una secuencia de caracteres intercalados, que represente las secuencias de datos concatenados de todas las secuencias de datos no numéricos del objeto empaquetado, analizando mediante el FinalCharacterMap, y:
- Para cada par de bits "00" encontrado en FinalCharacterMap, copie el siguiente valor de DigitVals a InterleavedString (agregue 48 a cada valor para convertirlo a ASCII);
- Para cada par de bits '01' encontrado en FinalCharacterMap, obtenga el siguiente valor de ExtDigitVals y use la Tabla K-2 para convertir ese valor a ASCII (o, si el valor es un cambio de Base 13, incremente más allá del siguiente par "01" 'en el FinalCharacterMap, y utilice ese valor de cambio de Base 13 más el siguiente valor Base 13 de ExtDigitVals para convertir el par de valores a ASCII). Almacene el resultado en InterleavedString;
- Para cada par de bits "10" encontrado en FinalCharacterMap, obtenga el siguiente carácter de NonDigitVals, convierta su valor base a un valor ASCII usando el Anexo K, y almacene el valor ASCII resultante en InterleavedString. Obtenga y procese un valor de Base 30 adicional por cada valor de cambio de Base 30 encontrado, para crear y almacenar un solo carácter ASCII.

- Para cada par de bits “11” encontrado en FinalCharacterMap, obtenga el siguiente carácter de ExtNonDigitVals, convierta su valor base a un valor ASCII usando el Anexo *K*, y almacene el valor ASCII resultante en InterleavedString, procesando cualquier Cambio como se describió anteriormente.

Una vez que se ha analizado el FinalCharacterMap por completo, InterleavedString se completa en su totalidad. A partir de la primera entrada alfanumérica en la lista de ID, copie los caracteres de InterleavedString a cada una de esas entradas, finalizando cada operación de copia después del número de caracteres indicado por los bits de longitud del formato auxiliar correspondiente, o al final de InterleavedString, lo que ocurra primero.

N Reconocimientos

Nombre	Empresa
Pete Alvarez	GS1 Oficina Global
Karen Arkesteyn	GS1 Bélgica y Luxemburgo
Xavier Barras	GS1 Francia
Jonas Buskenfried	GS1 Suecia
Kevin Dean	GS1 Canadá
Raymond Delnicki	GS1 EE. UU.
Sean Dennison	GS1 Irlanda
Vera Feuerstein	Nestlé
Richard Fisher	DoD Logistics AIT Oficina de estándares
Jean Christophe Gilbert	GS1 Francia
Ginger Green	Wal-Mart Stores, Inc.
Karolin Harsanji	GS1 Suecia
Yoshihiko Iwasaki	GS1 Japón
Steven Keddie	GS1 Oficina Global
Kimmo Keravuori	GS1 Finlandia
Ildiko Lieber	GS1 Hungría
Ilka Machemer	GS1 Alemania
Dan Mullen	GS1 Oficina Global
John Pearce	Axicon Auto ID Ltd
Sarina Pielaat	GS1 Países Bajos
Neil Piper	GS1 RU
Craig Alan Repec	GS1 Oficina Global
John Ryu	GS1 Oficina Global
Sue Schmid	GS1 Australia
Eugen Sehorz	GS1 Austria
Steven Simske	Colorado State University
Marie Vans	HP Inc.
Jane Wulff	GS1 Dinamarca