



UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

X.667

(09/2004)

SERIE X: REDES DE DATOS, COMUNICACIONES DE
SISTEMAS ABIERTOS Y SEGURIDAD

Gestión de redes de interconexión de sistemas abiertos y
aspectos de sistemas – Denominación, direccionamiento
y registro

Tecnología de la información – Interconexión de
sistemas abiertos – Procedimientos para el
funcionamiento de autoridades de registro OSI:
Generación y registro de identificadores únicos
universales y su utilización como componentes de
identificador de objetos ASN.1

Recomendación UIT-T X.667

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS, COMUNICACIONES DE SISTEMAS ABIERTOS Y SEGURIDAD

REDES PÚBLICAS DE DATOS	
Servicios y facilidades	X.1–X.19
Interfaces	X.20–X.49
Transmisión, señalización y conmutación	X.50–X.89
Aspectos de redes	X.90–X.149
Mantenimiento	X.150–X.179
Disposiciones administrativas	X.180–X.199
INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Modelo y notación	X.200–X.209
Definiciones de los servicios	X.210–X.219
Especificaciones de los protocolos en modo conexión	X.220–X.229
Especificaciones de los protocolos en modo sin conexión	X.230–X.239
Formularios para declaraciones de conformidad de implementación de protocolo	X.240–X.259
Identificación de protocolos	X.260–X.269
Protocolos de seguridad	X.270–X.279
Objetos gestionados de capa	X.280–X.289
Pruebas de conformidad	X.290–X.299
INTERFUNCIONAMIENTO ENTRE REDES	
Generalidades	X.300–X.349
Sistemas de transmisión de datos por satélite	X.350–X.369
Redes basadas en el protocolo Internet	X.370–X.379
SISTEMAS DE TRATAMIENTO DE MENSAJES	X.400–X.499
DIRECTORIO	X.500–X.599
GESTIÓN DE REDES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS Y ASPECTOS DE SISTEMAS	
Gestión de redes	X.600–X.629
Eficacia	X.630–X.639
Calidad de servicio	X.640–X.649
Denominación, direccionamiento y registro	X.650–X.679
Notación de sintaxis abstracta uno	X.680–X.699
GESTIÓN DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Marco y arquitectura de la gestión de sistemas	X.700–X.709
Servicio y protocolo de comunicación de gestión	X.710–X.719
Estructura de la información de gestión	X.720–X.729
Funciones de gestión y funciones de arquitectura de gestión distribuida abierta	X.730–X.799
SEGURIDAD	X.800–X.849
APLICACIONES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Compromiso, concurrencia y recuperación	X.850–X.859
Procesamiento de transacciones	X.860–X.879
Operaciones a distancia	X.880–X.899
PROCESAMIENTO DISTRIBUIDO ABIERTO	X.900–X.999
SEGURIDAD DE LAS TELECOMUNICACIONES	X.1000–

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

**Tecnología de la información – Interconexión de sistemas abiertos –
Procedimientos para el funcionamiento de autoridades de registro OSI:
Generación y registro de identificadores únicos universales
y su utilización como componentes de identificador de objetos ASN.1**

Resumen

En esta Recomendación | Norma Internacional se especifican los procedimientos para la generación y registro de identificadores únicos universales (UUID) y su utilización como componentes de identificador de objetos (OID) ASN.1 en el arco `{joint-iso-itu-t uuid(25)}`.

Orígenes

La Recomendación UIT-T X.667 fue aprobada el 13 de septiembre de 2004 por la Comisión de Estudio 17 (2001-2004) del UIT-T por el procedimiento de la Recomendación UIT-T A.8. Se publica también un texto idéntico como Norma Internacional ISO/CEI 9834-8.

PREFACIO

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

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

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

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

NOTA

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

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

PROPIEDAD INTELECTUAL

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

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

© UIT 2005

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

ÍNDICE

Página

1	Alcance.....	1
2	Referencias normativas	1
2.1	Recomendaciones Normas Internacionales idénticas	1
2.2	Otras referencias normativas.....	1
3	Términos y definiciones	2
3.1	Notación ASN.1	2
3.2	Autoridades de registro	2
3.3	Términos de red.....	2
3.4	Definiciones adicionales	2
4	Siglas.....	2
5	Notación	3
6	Estructura y representación del UUID.....	3
6.1	Estructura de campos del UUID.....	3
6.2	Representación binaria	4
6.3	Representación como un solo valor entero.....	4
6.4	Representación hexadecimal	4
6.5	Sintaxis formal de la representación hexadecimal	4
7	Utilización de un UUID para formar un OID.....	5
8	Utilización de un UUID para formar un URN.....	5
9	Reglas para comparar y ordenar los UUID.....	5
10	Validación	6
11	Bits de variante.....	6
12	Utilización de los campos de UUID y orden de los bytes en transmisión	6
12.1	Aspectos generales	6
12.2	Versión (<i>Version</i>)	7
12.3	Tiempo (<i>Time</i>).....	8
12.4	Secuencia de reloj (<i>Clock Sequence</i>).....	8
12.5	Nodo (<i>Node</i>).....	8
13	Configuración de los campos de un UUID basado en tiempo	9
14	Configuración de los campos de un UUID basado en nombre.....	9
15	Configuración de los campos de un UUID basado en número aleatorio	10
16	Registro de los UUID y utilización como componentes del OID.....	10
16.1	El árbol OID ASN.1	10
16.2	Nombramiento de las autoridades de registro	11
16.3	Tasas	11
16.4	Procedimientos de registro	11
16.4.1	Solicitud de registro de un UUID.....	11
16.4.2	Proceso de confirmación	11
16.4.3	Contenido del formulario	12
16.5	Mantenimiento del registro en la web	12
Anexo A	Algoritmos para la generación eficaz de UUID basados en tiempo	13
A.1	Algoritmo básico	13
A.2	Lectura del dispositivo de almacenamiento estable	13
A.3	Resolución del reloj de sistema	13
A.4	Escritura en el dispositivo de almacenamiento estable	14
A.5	Compartición de estado entre procesos	14
Anexo B	Propiedades de los UUID basados en nombre.....	15
Anexo C	Generación de números aleatorios en un sistema	16

	<i>Página</i>
Anexo D – Ejemplo	17
D.1 Ficheros proporcionados	17
D.2 Fichero <code>copyrt.h</code>	17
D.3 Fichero <code>uuid.h</code>	17
D.4 Fichero <code>uuid.c</code>	18
D.5 Fichero <code>sysdep.h</code>	21
D.6 Fichero <code>sysdep.c</code>	21
D.7 Fichero <code>utest.c</code>	23
D.8 Ejemplo de salida de <code>utest</code>	23
D.9 Algunos ID de espacio de nombre	24
BIBLIOGRAFÍA	25

Introducción

Esta Recomendación | Norma Internacional normaliza la generación y registro opcional de los identificadores únicos universales (UUID, *universally unique identifiers*).

Los UUID son cadenas de 16 octetos (128 bits). Los 16 octetos pueden interpretarse como una codificación de enteros sin signo y el valor entero resultante puede utilizarse como un arco del árbol OID en el arco `{joint-iso-itu-t uuid(25)}`, lo que permite a los usuarios generar OID sin someterse a un procedimiento de registro.

Los UUID también se denominan identificadores globalmente únicos (GUID, *globally unique identifiers*), pero este término no se utiliza en la presente Recomendación | Norma Internacional. En un principio, los UUID se usaron en el sistema computacional de redes (NCS) [1] y, posteriormente, en el entorno informático cómputo distribuido de la Open Software Foundation (DCE) [2]. En ISO/CEI 11578 [3] puede encontrarse una breve definición de algunos (aunque no todos) formatos UUID especificados en esta Recomendación | Norma Internacional. La especificación aquí contenida es coherente con todas las especificaciones anteriores.

Los UUID que forman un componente de un OID se representan en la notación de valor ASN.1 como una representación decimal de un valor entero, pero para otro tipo de representaciones, es más usual no representarlos con dígitos hexadecimales con un guión que separa los distintos campos dentro del UUID de 16 octetos. Esta representación se define en la presente Recomendación | Norma Internacional.

Si se generan de acuerdo con uno de los mecanismos definidos en esta Recomendación | Norma Internacional, se garantiza que los UUID son distintos de los UUID generados antes de 3603 A.D., o es muy probable que sean distintos (dependiendo del mecanismo elegido).

No se requiere una autoridad centralizada para administrar los UUID, pero se prevé el registro centralizado de los UUID autogenerados, de su generación automática (utilizando el algoritmo definido en esta Recomendación | Norma Internacional) y registro. Se garantiza que los UUID generados de manera centralizada son distintos de todos los UUID generados de manera centralizada. Se garantiza que los UUID registrados son distintos de todos los demás UUID registrados.

Un UUID puede servir a múltiples propósitos, desde etiquetar objetos de duración extremadamente corta hasta identificar fiablemente objetos muy duraderos en la red, en particular (aunque no necesariamente) como parte de un valor identificador de objeto (OID) ASN.1 o en un nombre de recurso uniforme (URN, *uniform resource name*).

El algoritmo de generación de UUID que se especifica en esta Recomendación | Norma Internacional soporta velocidades de asignación muy altas, de hasta 10 millones por segundo y por máquina, de ser necesario, por lo que es posible también utilizar los UUID como ID de transacción. En el anexo informativo se presenta un programa en lenguaje C que generará UUID de conformidad con esta Recomendación | Norma Internacional.

Se especifican tres algoritmos de generación de UUID únicos, que utilizan distintos mecanismos para garantizar su exclusividad. Estos algoritmos producen distintas versiones de un UUID.

El primer (y más común) mecanismo da como resultado la versión denominada basada en tiempo. Estos UUID pueden generarse a una velocidad de 10 millones por segundo. Para los UUID generados en un solo sistema informático, se utiliza una indicación de tiempo de 60 bits (utilizado como valor de reloj) con una granularidad de 100 nanosegundos, basado en el Tiempo Universal Coordinado (UTC) para garantizar la exclusividad durante un periodo de aproximadamente 1600 años. Para los UUID generados con la misma indicación de tiempo pero por distintos sistemas, la exclusividad se consigue utilizando direcciones de control de acceso a medios (MAC) de 48 bits, como se especifica en ISO/CEI 8802-3 (que se utiliza como valor Node). (Estas direcciones suelen estar disponibles en la mayoría de sistemas de red, pero, en caso contrario, pueden obtenerse de la autoridad de registro IEEE para las direcciones MAC – véase [4].) Se especifican otras maneras de generar valores de reloj y nodo para la versión basada en tiempo, si un sistema no dispone del tiempo UTC o si no hay una dirección MAC disponible.

El segundo mecanismo produce un único UUID en su versión basada en nombre, donde se utiliza el troceo criptográfico para obtener un valor UUID de 128 bits a partir de un nombre (de texto) inequívoco global.

El tercer mecanismo utiliza la generación de números aleatorios o pseudoaleatorios para producir la mayor parte de los bits de un valor de 128 bits.

La cláusula 5 especifica la notación utilizada para la denominación por orden de octetos y por orden de bits, y para especificar el orden de transmisión.

La cláusula 6 especifica la estructura de un UUID y su representación como valor binario, hexadecimal o entero simple.

En las cláusulas 7 y 8 se especifica la utilización de un UUID en un OID o un URN, respectivamente.

La cláusula 9 especifica las reglas de comparación de UUID para probar la igualdad o establecer una relación de orden entre dos UUID.

En la cláusula 10 se examina la posibilidad de comprobar la validez de un UUID. En general, los UUID tienen poca redundancia y hay poco margen para verificar su validez. No obstante, si se acepta un UUID para un registro, se garantiza que es distinto de todos los demás UUID registrados.

La cláusula 11 describe el uso histórico de algunos bits del UUID para definir variantes del formato UUID, y especifican el valor de estos bits en los UUID definidos de conformidad con la presente Recomendación | Norma Internacional.

En la cláusula 12 se especifica el uso de los campos de un UUID en las distintas versiones definidas (basada en el tiempo, basada en el nombre y basada en número aleatorio). También se define el orden de octetos en transmisión.

En la cláusula 13 se especifica la configuración de campos de un UUID basado en el tiempo.

En la cláusula 14 se especifica la configuración de campos de un UUID basado en el nombre.

En la cláusula 15 se especifica la configuración de campos de un UUID basado en un número aleatorio.

La cláusula 16 trata del funcionamiento de la autoridad de registro de los UUID que permite el registro centralizado y garantiza la exclusividad.

Todos los anexos son informativos.

El anexo A describe diversos algoritmos para el funcionamiento eficaz de los UUID basados en el tiempo.

En el anexo B se exponen las propiedades que debe tener un UUID basado en el nombre, que afectan a la selección de los espacios de nombre que se utilizan para generar dichos UUID.

En el anexo C se dan orientaciones sobre los mecanismos que pueden utilizarse para generar números aleatorios en un sistema informático.

El anexo D contiene un programa completo en lenguaje de programación C, que se puede utilizar para generar UUID.

**NORMA INTERNACIONAL
RECOMENDACIÓN UIT-T**

**Tecnología de la información – Interconexión de sistemas abiertos –
Procedimientos para el funcionamiento de autoridades de registro OSI:
Generación y registro de identificadores únicos universales
y su utilización como componentes de identificador de objetos ASN.1**

1 Alcance

La presente Recomendación | Norma Internacional especifica el formato y las reglas de generación que permiten a los usuarios producir identificadores de 128 bits que tienen la garantía de ser únicos globalmente o una gran probabilidad de serlo.

Los UUID generados de conformidad con la presente Recomendación | Norma Internacional son adecuados para un uso transitorio, ya que se genera un nuevo UUID cada 100 nanosegundos, o como identificadores permanentes.

Esta Recomendación | Norma Internacional está derivada de especificaciones no normalizadas anteriores de los UUID y su generación, y es técnicamente idéntica a dichas especificaciones.

Esta Recomendación | Norma Internacional especifica los procedimientos para el funcionamiento de una autoridad de registro de los UUID en la web.

Esta Recomendación | Norma Internacional también especifica y permite la utilización de los UUID (registrados o no registrados) como componentes OID en el arco `{joint-iso-itu-t uuid(25)}`, lo que permite a los usuarios generar OID sin someterse a procedimientos de registro.

En esta Recomendación | Norma Internacional también se especifica y permite la utilización de UUID (registrado o no registrados) para formar un URN.

2 Referencias normativas

Las siguientes Recomendaciones y Normas Internacionales contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de esta Recomendación | Norma Internacional. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y Normas son objeto de revisiones, por lo que se preconiza que los participantes en acuerdos basados en la presente Recomendación | Norma Internacional investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y las Normas citadas a continuación. Los miembros de la CEI y de la ISO mantienen registros de las Normas Internacionales actualmente vigentes. La Oficina de Normalización de las Telecomunicaciones de la UIT mantiene una lista de las Recomendaciones UIT-T actualmente vigentes.

2.1 Recomendaciones | Normas Internacionales idénticas

- Recomendación UIT-T X.660 (2004) | ISO/CEI 9834-1:2005, *Tecnología de la información – Interconexión de sistemas abiertos – Procedimientos para la operación de autoridades de registro para interconexión de sistemas abiertos: Procedimientos generales y arcos superiores del árbol de identificadores de objetos de ASN.1.*
- Recomendación UIT-T X.680 (2002) | ISO/CEI 8824-1:2002, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de la notación básica.*

2.2 Otras referencias normativas

- ISO/CEI 8802-3:2000, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.*
- ISO/CEI 10118-3:2004, *Information technology – Security techniques – Hash functions – Part 3: Dedicated hash-functions.*
- ISO/CEI 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS).*
- FIPS PUB 180-2:2002, *Federal Information Processing Standards Publication, Secure Hash Standard (SHS).*

- IETF RFC 1321 (1992), *The MD5 Message-Digest Algorithm*.
- IETF RFC 2141 (1997), *URN Syntax*.

3 Términos y definiciones

A los efectos de esta Recomendación | Norma Internacional, se aplican las siguientes definiciones.

3.1 Notación ASN.1

La presente Recomendación | Norma Internacional utiliza los siguientes términos, definidos en la Rec. UIT-T X.680 | ISO/CEI 8824-1:

- a) Tiempo Universal Coordinado (UTC, *Coordinated Universal Time*).
- b) Identificador de objeto (ASN.1).

3.2 Autoridades de registro

La presente Recomendación | Norma internacional utiliza los siguientes términos, definidos en la Rec. UIT-T X.660 | ISO/CEI 9834-1:

- a) árbol identificador de objeto (o árbol OID);
- b) registro;
- c) autoridad de registro;
- d) procedimientos de registro.

3.3 Términos de red

Esta Recomendación | Norma Internacional utiliza el siguiente término, definido en ISO/CEI 8802-3:

- Dirección MAC.

3.4 Definiciones adicionales

3.4.1 número aleatorio de calidad criptográfica (*cryptographic-quality random-number*): Número aleatorio o número pseudoaleatorio generado por un mecanismo que garantiza una separación suficiente de valores generados repetidamente para que sean aceptables para su uso en criptografía (y que se utilizan efectivamente).

3.4.2 versión basada en el nombre: UUID generado utilizando el troceo criptográfico de un nombre de espacio de nombre y de un nombre en dicho espacio de nombre.

3.4.3 espacio de nombre: Sistema para generar nombres de objeto que garantiza una identificación inequívoca dentro del espacio de nombre.

NOTA – Ejemplos de espacios de nombre son el sistema de nombre de dominio de red, los URN, los OID, los nombres distinguidos de directorio (véase[5]) y las palabras reservadas en un lenguaje de programación.

3.4.4 versión basada en número aleatorio: UUID generado utilizando un número aleatorio o pseudoaleatorio.

3.4.5 variante UUID normalizada: Una de las variantes de formatos UUID que se especifican en esta Recomendación | Norma Internacional:

NOTA – Tradicionalmente, ha habido otras especificaciones de formatos UUID que difieren de la variante especificada en esta Recomendación | Norma Internacional. Los UUID generados de acuerdo con estos distintos formatos son todos diferentes.

3.4.6 versión basada en el tiempo: UUID cuya exclusividad se obtiene utilizando una dirección MAC para identificar un sistema y un valor de reloj basado en el tiempo UTC actual.

3.4.7 identificador único universal (UUID, *universally unique identifier*): Valor de 128 bits generado de conformidad con esta Recomendación | Norma Internacional, o de acuerdo con otras especificaciones anteriores y que proporciona valores únicos entre sistemas y a lo largo del tiempo (véase igualmente 3.4.5).

4 Siglas

A los efectos de esta Recomendación | Norma Internacional, se utilizan las siguientes siglas.

ASN.1 Notación de sintaxis abstracta uno (*abstract syntax notation one*)

GUID	Identificador globalmente único (<i>globally unique identifier</i>)
IEEE	Institute of Electrical and Electronics Engineers, Inc.
MAC	Control de acceso a medios (<i>media access control</i>)
MD5	Message Digest algorithm 5
OID	Identificador de objeto ASN. 1 (<i>ASN.1 object identifier</i>)
RA	Autoridad de registro (<i>registration authority</i>)
SHA-1	Algoritmo de troceo seguro 1 (<i>secure hash algorithm 1</i>)
URL	Localizador de recurso uniforme (<i>uniform resource locator</i>)
URN	Nombre de recurso uniforme (<i>uniform resource name</i>)
UTC	Tiempo Universal Coordinado (<i>coordinated universal time</i>)
UUID	Identificador único universal (<i>universally unique identifier</i>)

5 Notación

5.1 En esta Recomendación | Norma Internacional se especifica una secuencia de octetos para un UUID utilizando los términos primero y último. El primer octeto también se denomina "octeto 15" y el último, "octeto 0".

5.2 Los bits dentro de un UUID también se numeran de "bit 127" a "bit 0", siendo el bit 127 el más significativo del octeto 15 y el bit 0 el menos significativo del octeto 0.

5.3 Cuando se utilizan figuras y cuadros en esta Recomendación | Norma Internacional, el octeto más significativo (y el bit más significativo) se muestran a la izquierda de la página. Esto corresponde con el orden de transmisión de octetos, donde los octetos más a la izquierda se transmiten en primer lugar.

5.4 Determinados valores utilizados en esta Especificación se expresan como el valor de un entero sin signo de una determinada longitud de bit (por ejemplo, N). Los bits del valor entero sin signo del bit N se numeran de "bit N-1" a "bit 0", siendo el bit N-1 el bit más significativo y el bit 0 el bit menos significativo.

5.5 Estas notaciones se utilizan exclusivamente a los efectos de esta Especificación. Las representaciones en las memorias informáticas no están normalizadas y dependen de la arquitectura del sistema.

6 Estructura y representación del UUID

6.1 Estructura de campos del UUID

6.1.1 Un UUID se especifica como una secuencia ordenada de seis campos. El UUID se especifica en términos de concatenación de estos campos UUID. Los campos UUID se denominan:

- a) campo "TimeLow";
- b) campo "TimeMid";
- c) campo "VersionAndTimeHigh";
- d) campo "VariantAndClockSeqHigh";
- e) campo "ClockSeqLow";
- f) campo "Node".

6.1.2 Los campos UUID se definen en orden de importancia, de acuerdo con la lista anterior, siendo "TimeLow" el campo más significativo (el bit 31 de "TimeLow" es el bit 127 del UUID), y el campo "Node" el campo menos significativo (el bit 0 de "Node" es el bit 0 del UUID).

6.1.3 Los contenidos de estos campos UUID se especifican en términos de un valor entero sin signo versión (Version), variante (Variant), tiempo, (Time) secuencia de reloj (Clock Sequence) y nodo (Node) (cada uno de los cuales tiene un tamaño de bits fijo). La configuración de estos valores se especifica en la cláusula 12 y su correspondencia con los anteriores campos UUID en 12.1.

NOTA – Como parte de los nombres de algunos de los campos UUID (por ejemplo, TimeLow, TimeMid, y TimeHigh) implican, el orden secuencial de los bits en un UUID (bit 127 a bit 0) que se deriva de un determinado valor entero sin signo (por ejemplo, bit 59 a 0 del valor Time) no es idéntico al orden secuencial de los bits en dicho valor entero sin signo, por motivos históricos.

6.2 Representación binaria

6.2.1 Un UUID se representará en notación binaria como 16 octetos formados por la concatenación de la codificación de enteros sin signo de longitud fija de cada uno de los campos en uno o más octetos. El número de octetos que se utiliza para cada campo será:

- a) campo "TimeLow": cuatro octetos;
- b) campo "TimeMid": dos octetos;
- c) campo "VersionAndTimeHigh": dos octetos;
- d) campo "VariantAndClockSeqHigh": un octeto;
- e) campo "ClockSeqLow": un octeto;
- f) campo "Node": seis octetos.

NOTA – Este orden de los campos UUID indica la representación habitual en un sistema informático y en la representación textual hexadecimal (véase 6.4).

6.2.2 El bit más significativo de la codificación de enteros sin signo de cada campo UUID será el bit más significativo del primer octeto (octeto N, octeto más significativo) y el bit menos significativo de la codificación de enteros sin signo será el bit menos significativo del último octeto (octeto 0, octeto menos significativo).

6.2.3 Los campos UUID estarán concatenados según su orden de importancia (véase 6.1.2), siendo el campo más significativo el primero y el campo menos significativo el último.

6.3 Representación como un solo valor entero

Un UUID puede representarse como un solo valor entero. Para obtener este solo valor entero del UUID, los 16 octetos de la representación binaria se considerarán una codificación de enteros sin signo siendo el bit más significativo de la codificación de enteros el bit más significativo (bit 7) del primero de los 16 octetos (octeto 15) y el bit menos significativo, el bit menos significativo (bit 0) del último de los 16 octetos (octeto 0).

NOTA – El valor entero único se utiliza cuando el UUID forma parte de un OID, como se especifica en la cláusula 7.

6.4 Representación hexadecimal

En el formato hexadecimal, los octetos del formato binario se representarán mediante una cadena de dígitos hexadecimales, utilizando dos dígitos hexadecimales para cada octeto del formato binario, siendo el primero el valor de los cuatro bits de orden superior del octeto 15, el segundo, el valor de los cuatro bits de orden inferior del octeto 15, etc., y siendo el último, el valor de los bits de orden inferior del octeto 0 (véase 6.5). Se insertará un carácter SÍMBOLO MENOS (45) (véase ISO/CEI 10646) entre las representaciones hexadecimales de cada par de campos adyacentes, excepto entre el campo "VariantAndClockSeqHigh" y el campo "ClockSeqLow" (véase el ejemplo en la cláusula 8).

6.5 Sintaxis formal de la representación hexadecimal

6.5.1 La definición formal de la sintaxis de representación hexadecimal de UUID se especifica utilizando la notación BNF extendida que se define en la Rec. UIT-T X.680 | ISO/CEI 8824-1, cláusula 5, excepto que no habrá espacios blancos entre los elementos léxicos.

6.5.2 El elemento léxico "hexdigit" se utiliza en la especificación BNF y se define de la siguiente manera:

Nombre del elemento léxico – hexdigit

Un "hexdigit" consistirá exactamente en uno de los siguientes caracteres:

A B C D E F a b c d e f 0 1 2 3 4 5 6 7 8 9

6.5.3 La representación hexadecimal de un UUID será la producción "UUID":

```

UUID ::=
    TimeLow
    " - " TimeMid
    " - " VersionAndTimeHigh
    " - " VariantAndClockSeqHigh ClockSeqLow
    " - " Node

TimeLow ::=
    HexOctet HexOctet HexOctet HexOctet

```

TimeMid ::=
 HexOctet HexOctet
VersionAndTimeHigh ::=
 HexOctet HexOctet
VariantAndClockSeqHigh ::=
 HexOctet
ClockSeqLow ::=
 HexOctet
Node ::=
 HexOctet HexOctet HexOctet HexOctet HexOctet HexOctet
HexOctet ::=
 hexdigit hexdigit

6.5.4 El soporte lógico que genere una representación hexadecimal de un UUID no utilizará letras mayúsculas.

NOTA – Se recomienda que la representación hexadecimal utilizada en todos los formatos legibles por personas se restrinja a letras minúsculas. El soporte lógico que procese esta representación deberá, no obstante, aceptar letras mayúsculas y minúsculas, como se especifica en 6.5.2.

7 Utilización de un UUID para formar un OID

Un OID formado con un UUID será:

{joint-iso-itu-t uuid(25) <uuid-single-integer-value>}

donde **<uuid-single-integer-value>** es el valor entero único del UUID que se especifica en 6.3.

NOTA – En la cláusula 16 se dan más detalles sobre la utilización de un UUID para formar un OID, y en 16.1.3 se dan orientaciones importantes sobre la exclusividad de los OID generados de esta manera.

8 Utilización de un UUID para formar un URN

Un URN (véase IETF RFC 2141) formado utilizando un UUID será una cadena **"urn:uuid:"** seguida de una representación hexadecimal del UUID, como se define en 6.4.

EJEMPLO – A continuación se presenta un ejemplo de la representación de cadena de un UUID como URN:

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

NOTA – Se dispone de un formato URN alternativo (véase [6]), pero no se recomienda para los URN generados utilizando UUID. Este formato alternativo utiliza el valor entero único de UUID especificado en 6.3 y representa el ejemplo anterior como **"urn:oid:2.25.329800735698586629295641978511506172918"**.

9 Reglas para comparar y ordenar los UUID

9.1 Para comparar un par de UUID, se comparan los valores de campos correspondientes (véase 6.1) de cada UUID, por orden de importancia (véase 6.1.2). Dos UUID son iguales si, y sólo si, todos los campos correspondientes son iguales.

NOTA 1 – Este algoritmo para comparar dos UUID es equivalente a la comparación de los valores de las representaciones de entero único que se especifican en 6.3.

NOTA 2 – Esta comparación utiliza los campos físicos especificados en 6.1.1 y no los valores enumerados en 6.1.3 y que se especifican en la cláusula 12 (Time, Clock Sequence, Variant, Version y Node).

9.2 Se considera que un UUID es mayor que otro, si tiene un valor más alto en el campo más significativo en el que difieren.

9.3 En un orden lexicográfico de representación hexadecimal de los UUID (véase 6.4), el UUID más grande seguirá al más pequeño.

10 Validación

Además de determinar si los bits de variante están correctamente configurados, y que el valor Time utilizado en el UUID basado en tiempo es futuro (y por tanto aún no puede asignarse), no hay un mecanismo para determinar si un UUID es válido en términos reales, dado que pueden darse todos los valores posibles.

11 Bits de variante

11.1 Los bits de variante son los tres bits más significativos (bits 7, 6, y 5) del octeto 7, que es el octeto más significativo del campo "VariantAndClockSeqHigh".

11.2 Todos los UUID conformes a la presente Norma Internacional tendrán los bits de variante configurados de la siguiente manera: bit 7 del octeto 7 puesto a 1 y bit 6 del octeto 7 puesto a 0. El bit 5 del octeto 7 es el bit más significativo de Clock Sequence (secuencia de reloj) y se configurará de acuerdo con 12.4.

NOTA – El bit 5 se considera aquí un bit de variante porque su valor distingue formatos históricos. De manera estricta, no forma parte del valor de variante en esta Recomendación | Norma Internacional, que sólo utiliza dos bits para la variante.

11.3 El cuadro 1 muestra, a título informativo, la utilización de otros valores de los bits de variante.

Cuadro 1 – Utilización de los bits de variante

Bit 7	Bit 6	Bit 5	Descripción
0	–	–	Reservado para proporcionar compatibilidad con versiones anteriores de NCS
1	0	–	Variante especificada en esta Recomendación Norma Internacional
1	1	0	Reservado para proporcionar compatibilidad con versiones anteriores de Microsoft Corporation
1	1	1	Reservado para uso futuro en esta Recomendación Norma Internacional

12 Utilización de los campos de UUID y orden de los bytes en transmisión

12.1 Aspectos generales

12.1.1 El cuadro 2 indica la posición y resume el uso de los distintos campos UUID en la representación binaria.

Cuadro 2 – Posición y utilización de los campos de UUID

Campo	Octeto # en UUID	Descripción
"TimeLow"	15-12	Bits de inferior orden del valor Time (32 bits)
"TimeMid"	11-10	Bits intermedios del valor Time (16 bits)
"VersionAndTimeHigh"	9-8	Version (4 bits) seguido de los bits de orden superior del valor Time (12 bits)
"VariantAndClockSeqHigh"	7	Bits del valor Variant (2 bits) seguido de los bits de orden superior de Clock Sequence (6 bits)
"ClockSeqLow"	6	Bits de orden inferior de Clock Sequence (8 bits)
"Node"	5-0	Node (véase 12.5) (48 bits)

12.1.2 En la figura 1 se ilustra la posición de los campos UUID en la representación binaria.

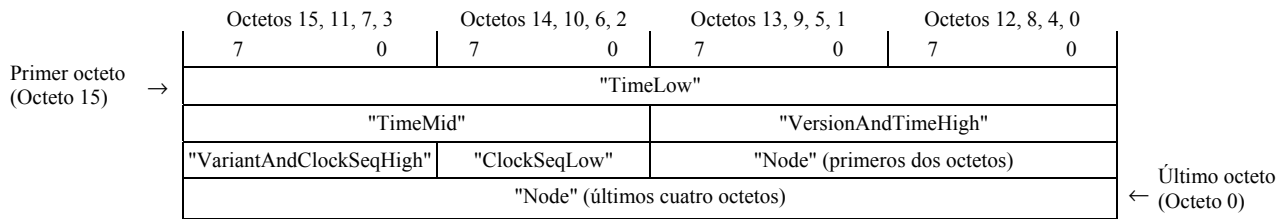


Figura 1 – Posición de los campos de UUID en la representación binaria

12.1.3 Se recomienda utilizar la representación binaria para la transmisión a través de un mecanismo de comunicación, transmitiendo los 16 octetos de la representación binaria como un conjunto contiguo de 16 octetos precediendo el primer octeto (octeto 15) al último octeto (octeto 0) durante la transmisión.

NOTA 1 – El orden de los bits dentro de un octeto está determinado por la especificación del mecanismo de comunicación.

NOTA 2 – Se recomienda utilizar 16 octetos consecutivos para la transmisión de un UUID, en el orden especificado anteriormente, pero las especificaciones de protocolo pueden elegir otros medios de transferir un UUID, incluida la fragmentación o la transmisión únicamente de porciones del UUID (como, por ejemplo, las partes que forman el valor Time).

12.2 Versión (*Version*)

12.2.1 Los tres medios alternativos de generación de un UUID (basado en tiempo, basado en nombre y basado en número aleatorio) se identifican y distinguen gracias a los cuatro bits más significativos del campo "VersionAndTimeHigh" (bits 7 a 4 del octeto 9 del UUID). Los UUID generados utilizando estos distintos mecanismos se denominan "distintas versiones de UUID".

NOTA – La denominación "distintas versiones de UUID" es algo confusa, pero se utiliza por motivos históricos. No existe tradicionalmente un concepto de "número de versión" para los formatos de UUID, donde las nuevas versiones pueden definirse como una revisión de la presente Recomendación | Norma Internacional. Cualquier nuevo formato UUID necesario en el futuro se identificará gracias a un valor distinto de los bits de variante.

12.2.2 En el cuadro 3 se enumeran las "versiones UUID" actualmente definidas, utilizando los primeros cuatro bits del campo "VersionAndTimeHigh" (bits 7 a 4 del octeto 9 del UUID). También se asigna un valor "Version" entero para cada combinación de bits.

NOTA – El valor de versión 2 no se utiliza por cuestiones de compatibilidad con definiciones históricas del UUID. Los valores de versión 0 y 6 a 15 se reservan para uso futuro.

Cuadro 3 – Versiones de UUID actualmente definidas

Bit 7	Bit 6	Bit 5	Bit 4	Valor de versión	Descripción
0	0	0	1	1	Versión basada en el tiempo especificada en esta Recomendación Norma Internacional (véase la cláusula 13)
0	0	1	0	2	Reservado para la versión de seguridad de DCE, con UUID POSIX incorporados
0	0	1	1	3	Versión basada en nombre especificada en esta Recomendación Norma Internacional con troceo MD5 (véase la cláusula 14)
0	1	0	0	4	Versión basada en un número aleatorio especificada en esta Recomendación Norma Internacional (véase la cláusula 15)
0	1	0	1	5	Versión basada en el nombre especificada en esta Recomendación Norma Internacional con troceo SHA-1 (véase la cláusula 14)

12.3 Tiempo (*Time*)

12.3.1 Time será un valor de 60 bits.

NOTA – El nombre "Time" es adecuado para la versión basada en tiempo de un UUID (versión 1), pero también se utiliza para el contenido del valor correspondiente en otras versiones de UUID (versiones 3 y 4).

12.3.2 Para la versión basada en tiempo de un UUID, Time será el cómputo de intervalos de 100 nanosegundos en Tiempo Universal Coordinado (UTC) desde la medianoche del 15 de octubre de 1582 (fecha de la reforma Gregoriana del Calendario Cristiano).

NOTA 1 – Antes de la creación de la Oficina Internacional de la Hora, los minutos contenían exactamente 60 segundos. Desde entonces, se han modificado los segundos cuando ha sido necesario, incrementando (o en ocasiones reduciendo) el número de segundos por año.

NOTA 2 – Los sistemas portátiles pueden tener problemas para determinar el tiempo UTC, ya que generalmente están fijados al tiempo local de su estación base. Aunque continúen utilizando la hora local de la base, o modifique el valor de Clock Sequence (véase 12.4), los UUID generados seguirán siendo únicos.

NOTA 3 – Para que los sistemas que no tengan acceso a la difusión de señales horarias, podrá utilizarse un sistema de reloj que registre el tiempo local con un diferencial añadido, siempre y cuando no se generen UUID en el periodo de cambio a la hora de verano, o cuando se realice un cambio del valor Clock Sequence (véase 12.4).

12.3.3 Para la versión basada en el nombre del UUID, éste será un valor de 60 bits construido a partir de un nombre globalmente único, como se especifica en la cláusula 14.

NOTA – Ejemplos de nombres globalmente únicos son OID, URN y nombres distinguidos del directorio (véase [5]).

12.3.4 Para la versión basada en un número aleatorio de un UUID, se tratará de un valor de 60 bits generado aleatoria o pseudoaleatoriamente, como se especifica en la cláusula 15.

12.4 Secuencia de reloj (*Clock Sequence*)

12.4.1 Para la versión basada en el tiempo del UUID, Clock Sequence se utiliza para evitar las duplicaciones que podrían ocurrir cuando el valor Time se retrasa o si se modifica el valor Node.

NOTA – El nombre "Clock Sequence" es adecuado para la versión basada en el tiempo del UUID, pero también se utiliza para el contenido del valor correspondiente en las versiones basada en el nombre y basada en un número aleatorio del UUID.

12.4.2 Si el valor Time se retrasa, o ha sido retrasado (por ejemplo, mientras el sistema estaba apagado), el generador de UUID no puede saber si un UUID ha sido generado con valores Time mayores que el valor al que está configurado. En estos casos, se deberá modificar el valor de Clock Sequence.

NOTA – Si se conoce el anterior valor de Clock Sequence, puede simplemente incrementarse. En cualquier otro caso, debe ponerse a un valor aleatorio o pseudoaleatorio de calidad criptográfica.

12.4.3 Del mismo modo, si se modifica el valor de Node (por ejemplo, porque la tarjeta de red se ha cambiado de una máquina a otra), deberá cambiarse el valor de Clock Sequence.

12.4.4 Clock Sequence deberá inicializarse en un principio (es decir, una vez durante la vida del sistema que produce los UUID) a un número aleatorio no derivado del valor Node.

NOTA – Esto se hace para minimizar la correlación entre sistemas, proporcionando una mayor protección contra las direcciones MAC que pueden cambiar o conmutar de un sistema a otro rápidamente.

12.4.5 Para la versión del UUID basada en nombre, Clock Sequence será un valor de 14 bits construido a partir de un nombre, como se especifica en la cláusula 14.

12.4.6 Para la versión del UUID basada en número aleatorio, Clock Sequence será un valor de 14 bits generado aleatoria o pseudoaleatoriamente, como se especifica en la cláusula 15.

12.5 Nodo (*Node*)

12.5.1 Para la versión basada en tiempo de un UUID, el valor Node estará formado por una dirección MAC (véase ISO/CEI 8802-3), que normalmente será la dirección anfitrión de alguna interfaz de red.

12.5.2 En los sistemas con múltiples direcciones MAC cabe utilizar cualquier dirección disponible, excepto una dirección de multidifusión. El octeto 5 del UUID (primer octeto de "Node") se pondrá al primer octeto de la dirección MAC que se transmite mediante un sistema conforme a la Norma ISO/CEI 8802-3.

NOTA 1 – Este octeto contiene el bit global/local y el bit unidifusión/multidifusión. Es necesario que el bit unidifusión/multidifusión esté puesto a unidifusión para evitar problemas con direcciones generadas de acuerdo con 12.5.3.

NOTA 2 – Es posible obtener un bloque de direcciones MAC a partir de la autoridad de registro de direcciones MAC (véase [4]).

12.5.3 Para los sistemas sin dirección MAC, podrá utilizarse un número aleatorio o pseudoaleatorio de calidad criptográfica (véase el anexo C). El bit de multidifusión deberá estar configurado en dichas direcciones.

NOTA – Se garantiza así que las direcciones generadas no entren en conflicto con direcciones obtenidas a partir de tarjetas de red, como se especifica en 12.5.2.

12.5.4 Para un UUID basado en nombre, el valor Node será un valor de 48 bits construido mediante canonización y troceo de un nombre globalmente único, como se especifica en la cláusula 14.

12.5.5 Para un UUID basado en número aleatorio, el valor Node será un valor de 48 bits generado aleatoria o pseudoaleatoriamente, como se especifica en la cláusula 15.

13 Configuración de los campos de un UUID basado en tiempo

Los campos de un UUID basado en tiempo se configurarán de la siguiente manera:

- Se determinarán los valores Time y Clock Sequence de acuerdo con el UTC, que se utilizarán en el UUID, como se especifica en 12.3 y 12.4.
- A los efectos de este algoritmo, se considera que Time es un entero sin signo de 60 bits y Clock Sequence un entero sin signo de 14 bits. Se numerarán secuencialmente los bits de cada valor, utilizando el cero para el bit menos significativo.
- El campo "TimeLow" será igual a los 32 bits menos significativos (bits 31 a 0) de Time y en el mismo orden de importancia.
- El campo "TimeMid" será igual a los bits 47 a 32 de Time y en el mismo orden de importancia.
- Los 12 bits menos significativos (bits 11 a 0) del campo "VersionAndTimeHigh" serán iguales a los bits 59 a 48 de Time y en el mismo orden de importancia.
- Los cuatro bits más significativos (bits 15 a 12) del campo "VersionAndTimeHigh" serán iguales al número de versión de cuatro bits que se especifica en 12.2.
- El campo "ClockSeqLow" será igual a los ocho bits menos significativos (bits 7 a 0) de Clock Sequence y en el mismo orden de importancia.
- Los seis bits menos significativos (bits 5 a 0) de "VariantAndClockSeqHigh" serán iguales a los seis bits más significativos (bits 13 a 8) de Clock Sequence y en el mismo orden de importancia.
- Los dos bits más significativos (bits 7 y 6) del campo "VariantAndClockSeqHigh" se pondrán a uno y a cero, respectivamente.
- El campo Node será igual a la dirección MAC de 48 bits siguiendo el mismo orden de importancia.

14 Configuración de los campos de un UUID basado en nombre

En esta cláusula se especifican los procedimientos para la producción de un UUID basado en nombre. En la subcláusula 14.1 se describen los procedimientos generales de cualquier función de troceo (véase igualmente ISO/CEI 10118-3). En la subcláusula 14.2 se examina el uso de MD5, y en la subcláusula 14.3, el uso de SHA-1.

NOTA – La utilización de MD5 está restringida a aquellos casos que requieran compatibilidad con UUID ya existentes, ya que SHA-1 proporciona un algoritmo de troceo con menor probabilidad de que se produzca el mismo valor de troceo con otro tipo de algoritmos (véase C.4).

14.1 Los campos del UUID basado en nombre se configurarán de la siguiente manera:

- Se asignará un UUID para usarlo como "identificador de espacio de nombre" para todos los UUID generados a partir de nombres de dicho espacio de nombre.
NOTA – En D.9 recomiendan los UUID para los cuatro espacios de nombre más comúnmente utilizados.
- Se convertirá el nombre a una secuencia canónica de octetos (como se define en las normas o convenios de su espacio de nombre).
- Se computará el valor de troceo de 16 octetos del identificador de espacio de nombre concatenado con el nombre, utilizando la función de troceo especificada en 14.2 ó 14.3. La numeración de los octetos en el valor troceado va de 0 a 15, como se especifica en IETF RFC 1321 para el MD5, y como se indica en FIPS PUB 180-2 para SHA-1.
- Los octetos 3 a 0 del campo "TimeLow" serán iguales a los octetos 3 a 0 del valor de troceo.
- Los octetos 1 y 0 del campo "TimeMid" serán iguales a los octetos 5 y 4 del valor de troceo.
- Los octetos 1 y 0 del campo "VersionAndTimeHigh" serán iguales a los octetos 7 y 6 del valor de troceo.
- Se sobrescribirán los cuatro bits más significativos (bits 15 a 12) del campo "VersionAndTimeHigh" con el número de versión de cuatro bits del cuadro 3 de 12.2.2 para la función de troceo que se utilizó.

- El campo "VariantAndClockSeqHigh" será igual al octeto 8 del valor de troceo.
- Se sobrescribirán los dos bits más significativos (bits 7 y 6) del campo "VariantAndClockSeqHigh" con los valores 1 y 0, respectivamente.
- El campo "ClockSeqLow" será igual al octeto 9 del valor de troceo.
- Los octetos 5 a 0 del campo "Node" serán iguales a los octetos 15 a 10 del valor de troceo.

14.2 En esta subcláusula se especifica el UUID basado en nombre utilizando el MD5 como función de troceo, pero esta función no se utilizará para los UUID de nueva generación (véase C.4). Para la función de troceo MD5, el "valor de troceo" que se indica en 14.1 es el valor de 16 octetos especificado en IETF RFC 1321, como octetos 0 a 15.

NOTA – Esta especificación de MD5, con el número de versión asociado, se incluye únicamente para compatibilidad con versiones anteriores de la especificación.

14.3 En esta subcláusula se especifica el UUID basado en nombre utilizando SHA-1 como función de troceo. Para la función de troceo SHA-1, el "valor de troceo" que se indica en 14.1 serán los octetos 0 a 15 del valor de 20 octetos obtenido del valor Message Digest de 160 bits que se especifica en FIPS PUB 180-2. Se descartarán los octetos 16 a 19 del valor de 20 octetos. El valor de 20 octetos se obtendrá a partir del valor Message Digest de 160 bits de FIPS PUB 180-2 colocando el bit más significativo del valor de 160 bits en el bit más significativo del primer octeto (octeto 0) del valor de 20 octetos, y el bit menos significativo en el último octeto (octeto 19) del valor de 20 octetos.

15 Configuración de los campos de un UUID basado en número aleatorio

15.1 Los campos de un UUID basado en número aleatorio se configurarán de la siguiente manera:

- Los dos bits más significativos (bits 7 y 6) del campo "VariantAndClockSeqHigh" se pondrán a 1 y 0, respectivamente.
- Los cuatro bits más significativos (bits 15 a 12) del campo "VersionAndTimeHigh" serán iguales al número de versión de cuatro bits que se especifica en 12.2.
- Todos los demás bits del UUID se pondrán a valores generados aleatoriamente (o pseudoaleatoriamente).

NOTA – Los números pseudoaleatorios pueden originar el mismo valor múltiples veces. Se recomienda vivamente el uso de números aleatorios de calidad criptográfica para reducir la probabilidad de valores repetidos.

15.2 En el anexo C se dan orientaciones para generar números aleatorios en un sistema.

16 Registro de los UUID y utilización como componentes del OID

16.1 El árbol OID ASN.1

NOTA – En esta subcláusula se resumen las principales disposiciones de la Rec. UIT T X.660 | ISO/CEI 9834-1.

16.1.1 Esta Recomendación | Norma Internacional describe procedimientos para el funcionamiento de una autoridad de registro de UUID. Este registro también permite la utilización de dichos UUID como arcos en el árbol OID dentro del arco {`joint-iso-itu-t uuid(25)`} (véase también la cláusula 7).

NOTA – Los UUID pueden ser utilizados también para identificar arcos dentro del arco `uuid(25)` sin estar registrados, pero no se garantiza que la identificación de dichos arcos sea mundialmente inequívoca.

16.1.2 Los OID son una forma de identificación inequívoca mundial basada en una estructura de árbol jerárquica, independiente de las autoridades de registro jerárquicas. El árbol OID tiene un nodo raíz, arcos por debajo del nodo de raíz, arcos por debajo de dichos arcos, etc., ilimitadamente. Los arcos se identifican gracias a valores enteros no negativos (véase 16.1.5) que proporcionan la identificación inequívoca de un arco dentro del nodo superior. También pueden darse a los arcos nombres (formados por una o más letras minúsculas, letras mayúsculas, dígitos y guiones, con una letra minúscula inicial, sin guiones adyacentes y sin guión al final), pero están supeditados a valores numéricos y no son necesarios. Un objeto se identifica gracias a la secuencia de valores de arco (numéricos o, igualmente, arcos tempranos, nombres de arco) del nodo raíz al objeto.

NOTA – Para una descripción más detallada del árbol OID, véanse las Recomendaciones UIT-T X.660 | ISO/CEI 9834-1 y X.680 | ISO/CEI 8824-1.

16.1.3 (INFORMATIVO) Es importante señalar que es posible utilizar un UUID no registrado en el mismo arco que un UUID registrado (véase 16.1.1). Por consiguiente, se puede utilizar valores idénticos para un UUID registrado y un UUID no registrado (o para dos UUID no registrados), aunque la probabilidad de que esto ocurra es muy pequeña. Esta probabilidad aumenta si los UUID se generan a partir de valores de troceo MD5 o números pseudoaleatorios, en vez de a partir de valores de troceo SHA-1 y números aleatorios de calidad criptográfica. Esto puede causar confusión a los usuarios del OID, y dar pie a utilizaciones maliciosas, como la simulación. La autoridad de registro de los UUID es

responsable de cualquier coincidencia entre los UUID registrados, pero no es responsable de las coincidencias entre UUID registrados y UUID no registrados, ya que no gestiona los UUID no registrados. Si ocurre este tipo de coincidencia, la semántica asociada con el UUID registrado tendrá preferencia, y no deben utilizarse las semánticas asociadas con los UUID no registrados. Así, el registro de un OID con un UUID no garantiza que el OID sea más exclusivo que su UUID. El registro debe servir principalmente como medio de publicar los UUID y su semántica.

16.1.4 Es posible, en representaciones automáticas, que un OID implique (por el contexto de la representación en máquina) la identificación de la parte de un trayecto de la raíz al objeto en un árbol OID. Así pues, si se sabe que un OID está formado como se especifica en la cláusula 7, la representación en máquina consistirá únicamente en el valor UUID.

16.1.5 Los componentes de los OID son enteros no negativos de magnitud ilimitada.

16.2 Nombramiento de las autoridades de registro

16.2.1 Corresponde al mandato de la UIT | ISO/CEI organizar el registro como se especifica en esta Recomendación | Norma Internacional. Para ello, el UIT-T y la ISO/CEI nombran, de conformidad con sus requisitos y normas internas, una organización que ejerza de autoridad de registro en el marco de esta Recomendación | Norma Internacional.

NOTA – Para registro, utilícese el URL <http://www.itu.int/ITU-T/asn1/uuid.html> (sitio del Proyecto ASN.1 del UIT-T).

16.2.2 La autoridad de registro no será responsable de los fallos de funcionamiento de estos procedimientos o de cualquier acción relacionada con sus obligaciones en el marco de esta Recomendación | Norma Internacional, exceptuando que podrá ser eximida de sus funciones por la Comisión de Estudio del UIT-T o Subcomité de la ISO/CEI JTC 1 pertinentes, sin penalización. La autoridad de registro no se considera responsable de la utilización de cualquier valor OID no registrado idéntico a un valor OID registrado, ya que no tiene control alguno sobre la utilización de dichos valores (véase 16.1.3).

NOTA – Si la Comisión de Estudio del UIT-T o el Subcomité de la ISO/CEI JTC 1 pertinentes determinan que la autoridad de registro ha de ser eximida de sus funciones, por cualquier motivo que sea, se supone que la información de registro mantenida por la RA se pondrá a disposición de cualquier RA que se nombre posteriormente.

16.3 Tasas

16.3.1 La organización que ejerza como autoridad reguladora, lo hará en base al método de recuperación de costos. Se establecerá la estructura tarifaria para recuperar los gastos incurridos por la autoridad de registro, cubrir la publicación en la web de los registros y las peticiones de información, de manera que se limite su número y se realicen únicamente por motivos importantes.

16.3.2 La cuantía de las tasas será determinada por la autoridad de registro, previa aprobación de la Comisión de Estudio del UIT-T | Subcomité de la ISO/CEI JTC 1 pertinentes. Las tasas pueden aplicarse por:

- a) registro;
- b) petición de información;
- c) publicación en la web;
- d) petición de actualización.

16.3.3 Las tasas se aplicarán por igual, independientemente del país a partir del cual se realiza la solicitud y estarán sujetas a las fluctuaciones del tipo de cambio.

16.4 Procedimientos de registro

En esta subcláusula se especifican los procedimientos que habrán de seguirse para registrar un UUID, y que están previstos para garantizar la transparencia y el funcionamiento adecuado de la autoridad de registro.

16.4.1 Solicitud de registro de un UUID

16.4.1.1 Toda organización que presente una solicitud de registro de un UUID directamente a la RA, deberá rellenar el formulario que figura en la página web de la RA. El contenido de la solicitud se especifica en 16.4.3.

16.4.1.2 Una vez completados satisfactoriamente los procedimientos de registro, se registrará y publicará el valor de UUID de 128 bits que se ha sido asignado a (o por) la organización solicitante.

16.4.2 Proceso de confirmación

En el sitio web de la RA se confirman y publican los registros realizados satisfactoriamente.

16.4.3 Contenido del formulario

16.4.3.1 En esta cláusula se especifica la información que necesita la RA para efectuar el procedimiento de registro.

NOTA – En el momento de publicación de esta Recomendación | Norma Internacional, esta información puede presentarse por correo electrónico, teléfono, en papel o directamente en el sitio web de la RA.

16.4.3.2 El registro comprende la siguiente información:

- a) país donde se encuentra la sede de la organización registrante;
- b) nombre de la organización, información de registro del país, si se trata de una empresa, registrada, organización caritativa, etc., o afiliación a una organización internacional conocida;
- c) nombre y cargo, dirección postal, dirección de correo electrónico, teléfono y fax de la persona de contacto en la organización registrante;
- d) declaración de buenas intenciones de la organización registrante como medio para controlar y evitar los registros innecesarios;
- e) (opcional) URL accesible donde se pueda encontrar más información sobre la utilización del UUID.

16.4.3.3 El contenido del formulario general para un OID se especifica en la Rec. UIT-T X.660 | ISO/CEI 9834-1, cláusula 8.

16.5 Mantenimiento del registro en la web

16.5.1 La autoridad de registro mantendrá en un sitio web de su elección una lista de todos los registros efectuados.

16.5.2 La RA actualizará, libre de cargo, toda información sobre las organizaciones registradas si se comunican cambios del nombre de la empresa o información similar y la autorización correspondiente para efectuar el cambio. El mecanismo utilizado para ello será determinado por la RA y se indicará en su sitio web.

Anexo A

Algoritmos para la generación eficaz de UUID basados en tiempo

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

En este anexo se describe un algoritmo que puede utilizarse para generar repetidamente UUID basados en tiempo en un sistema informático.

A.1 Algoritmo básico

A.1.1 El siguiente algoritmo es simple, correcto, pero ineficaz:

- Obtener un bloqueo general del sistema.
- A partir de un dispositivo de almacenamiento estable compartido por todo el sistema (por ejemplo, un fichero), leer el estado del generador de UUID: valores de Time, Clock Sequence y Node utilizados para generar el último UUID.
- Obtener el tiempo actual como un cómputo de 60 bits de intervalos de 100 nanosegundos desde las 00:00:00:00 del 15 de octubre de 1582 y tomado como valor Time.
- Obtener el valor Node actual.
- Si el estado es no disponible (por ejemplo, no existe o está corrompido), o el valor de Node guardado es distinto del valor de Node actual, generar un valor Clock Sequence aleatorio.
- Si el estado es disponible, pero el valor Time guardado es posterior al valor Time actual, incrementar el valor Clock Sequence.
- Guardar el estado (valores Time, Clock Sequence y Node actuales) en el dispositivo de almacenamiento estable.
- Suprimir el bloqueo general.
- Formar un UUID a partir de los valores Time, Clock Sequence y Node actuales, de acuerdo con lo indicado en la cláusula 13.

A.1.2 Si no es necesario generar UUID con frecuencia, el algoritmo anterior puede ser perfectamente adecuado. No obstante, si se necesita un funcionamiento de mayor calidad, el algoritmo plantea los siguientes problemas:

- La lectura del estado a partir del dispositivo de almacenamiento estable a cada vez es ineficiente.
- La resolución del reloj de sistema puede no ser de 100 nanosegundos.
- La escritura del estado en el dispositivo de almacenamiento estable a cada vez es ineficiente.
- Puede ser ineficiente la compartición de estados entre fronteras del proceso.

A.1.3 Cada una de estas cuestiones puede ser tratada modularmente, realizando mejoras locales de las funciones de lectura y escritura del estado y de lectura del reloj, como se trata más adelante.

A.2 Lectura del dispositivo de almacenamiento estable

A.2.1 El estado sólo ha de leerse en el dispositivo de almacenamiento estable una vez en el momento de arranque, si se traslada a un dispositivo de almacenamiento volátil compartido del sistema (y se actualiza siempre que se actualiza el dispositivo de almacenamiento estable).

A.2.2 Si la implementación no dispone de un dispositivo de almacenamiento estable, siempre puede indicar que los valores no están disponibles. Estas implementaciones son las menos deseables, ya que se incrementa la frecuencia de creación de nuevos números de Clock Sequence, lo que aumenta la probabilidad de duplicación.

A.2.3 Si el valor Node no puede cambiar (por ejemplo, la tarjeta de red es inseparable del sistema), o si cualquier cambio reinicializa también Clock Sequence a un valor aleatorio, en vez de mantenerlo en el dispositivo de almacenamiento estable, puede devolverse el valor Node actual.

A.3 Resolución del reloj de sistema

A.3.1 El valor Time se genera a partir del tiempo del sistema, cuya resolución puede ser inferior a la resolución necesaria para Time.

A.3.2 Si no hay que generar UUID con frecuencia, Time puede simplemente ser el tiempo del sistema multiplicado por el número de intervalos de 100 nanosegundos por intervalo de tiempo del sistema.

A.3.3 Si un sistema desborda el generador pidiendo demasiados UUID dentro de un mismo intervalo de tiempo de sistema, el servicio UUID debe devolver un error o retener el generador UUID hasta que el reloj de sistema alcance su nivel.

A.3.4 Es posible simular un valor Time de alta resolución manteniendo el cómputo de cuántos UUID se han generado con el mismo valor de tiempo del sistema, y utilizándolo para construir los bits de orden inferior del valor Time. El cómputo oscilará entre cero y el número de intervalos de 100 nanosegundos de cada intervalo de tiempo del sistema.

NOTA – Si los procesadores desbordan frecuentemente la generación de UUID, podrán atribuirse al sistema direcciones MAC adicionales que permitirán una asignación más rápida multiplicando el número de UUID potencialmente disponibles para cada valor Time.

A.4 Escritura en el dispositivo de almacenamiento estable

No siempre ha de escribirse el estado en el dispositivo de almacenamiento estable cada vez que se genera un UUID. El valor Time en el dispositivo de almacenamiento estable puede ponerse periódicamente a un valor superior a cualquiera de los ya utilizados en un UUID. Mientras los UUID generados tengan valores Time inferiores a dicho valor, los valores Clock Sequence y Node permanecerán inalterados, y sólo habrá que actualizar la copia del estado en el dispositivo de almacenamiento volátil compartido. Además, si el valor Time en el dispositivo de almacenamiento estable es futuro por un valor menor que el tiempo típico que la reinicialización del sistema, cualquier coincidencia no causará la reinicialización de Clock Sequence.

A.5 Compartición de estado entre procesos

Si resulta demasiado oneroso acceder al estado compartido cada vez que se genera un UUID, el generador del sistema puede configurarse de manera que atribuya un bloque de valores Time cada vez que se le solicita, y un generador de proceso puede atribuir valores de dicho bloque hasta que se agote.

Anexo B

Propiedades de los UUID basados en nombre

(Este anexo no forma parte integrante de la presente Recomendación | Norma Internacional)

B.1 El UUID basado en nombre está destinado a generar un UUID a partir de un nombre que se extrae de un espacio de nombre y es único dentro de éste. El concepto de nombre y espacio de nombre debe considerarse en el sentido más amplio y no se limita a los nombres textuales. Los mecanismos o convenios para asignar nombres, a partir de sus espacios de nombre y garantizando su exclusividad dentro de éstos, quedan fuera del alcance de esta Recomendación.

NOTA – Para evitar problemas de repetición, los UUID basados en nombre no se generarán a partir de un OID que termine con un UUID que sea, a su vez, basado en nombre.

B.2 Las propiedades de los UUID basados en nombre generados de conformidad con la cláusula 14 y con un espacio de nombre adecuadamente elegido serán las siguientes:

- los UUID generados en distintos momentos a partir del mismo nombre y del mismo espacio de nombres serán iguales;
- los UUID generados a partir de dos nombres distintos en el mismo espacio de nombre serán, con gran probabilidad, distintos;
- los UUID generados a partir del mismo nombre en dos espacios de nombre distintos serán, con muy alta probabilidad, distintos;
- si dos UUID basados en nombre son iguales, quiere decir que con mucha probabilidad fueron generados a partir del mismo nombre en el mismo espacio de nombre.

Anexo C

Generación de números aleatorios en un sistema

(Este anexo no forma parte integrante de la presente Recomendación | Norma Internacional)

C.1 Aunque algunos sistemas no tienen la capacidad de generar números aleatorios de calidad criptográfica, en muchos de ellos se dispone por lo general de un número relativamente importante de fuentes de aleatoriedad a partir de las cuales puede generarse uno de esos números. Esas fuentes son específicas del sistema, pero suelen incluir:

- el porcentaje de memoria utilizada;
- la capacidad de memoria principal en bytes;
- la capacidad de memoria principal libre en bytes;
- el tamaño del fichero de desplazamiento por páginas en bytes;
- los bytes libres del fichero de desplazamiento por páginas;
- el tamaño total del espacio de la dirección virtual de usuario en bytes;
- los bytes totales disponibles del espacio de la dirección de usuario;
- el tamaño de la unidad del disco de carga en bytes;
- el espacio libre del disco de la unidad de carga en bytes;
- el tiempo actual;
- el tiempo transcurrido desde la carga inicial del sistema;
- las dimensiones de cada fichero en los diversos directorios del sistema;
- los tiempos de creación, última lectura y modificación de ficheros en los distintos directorios del sistema;
- los factores de utilización de los diversos recursos del sistema (heap, etc.);
- la posición actual del cursor del ratón;
- la posición actual del signo de intercalación;
- el número actual de procesos, aplicaciones en curso;
- iconos o ID en la ventana del escritorio y en la ventana activa;
- el valor del puntero de pila del llamante;
- el proceso de identificador del llamante;
- los diversos contadores de calidad de funcionamiento específicos de la arquitectura del procesador (instrucciones ejecutadas, fallos de cache, memoria intermedia de traducción, o fallos de ésta).

C.2 Además, elementos tales como el nombre de la computadora y el nombre del sistema operativo, aunque estrictamente hablando no sean aleatorios, pueden contribuir a diferenciar los resultados de los obtenidos por otros sistemas.

C.3 El algoritmo exacto para generar un valor Node utilizando estos datos es específico del sistema, ya que tanto los datos disponibles como las funciones para obtenerlos suelen ser específicos del sistema. No obstante, un método genérico, es acumular el mayor número posible de fuentes en una memoria intermedia y utilizar un Message Digest, como SHA-1, tomar arbitrariamente seis bytes del valor de troceo y configurar el bit de multidifusión como se describe anteriormente.

C.4 Es posible utilizar igualmente otras funciones de troceo, como MD5 y las funciones de troceo especificadas en ISO/CEI 10118. El único requisito es que el resultado sea adecuadamente aleatorio en el sentido de que las salidas obtenidas de un conjunto de entradas uniformemente distribuidas estén también uniformemente distribuidas y que el cambio de un solo bit en la entrada pueda causar el cambio de la mitad de los bits de salida. (No obstante, no se recomienda utilizar MD5 para los nuevos UUID ya que las últimas investigaciones muestran que sus valores de salida obtenidos no están distribuidos uniformemente.)

Anexo D

Ejemplo de implementación

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

D.1 Ficheros proporcionados

Esta implementación está formada por 6 ficheros: `copyrt.h`, `uuid.h`, `uuid.c`, `sysdep.h`, `sysdep.c` y `utest.c`. Los ficheros `uuid.*` son una implementación independiente del sistema de algoritmos de generación de UUID que se describen en las cláusulas 13, 14 y 15, con todas las mejoras que se incluyen en el anexo A (excepto la compartición eficiente del estado entre procesos). El código asume el soporte de enteros de 64 bits, lo que lo hace mucho más claro.

NOTA – Este código ha sido probado en Linux (Red Hat 4.0) con GCC (2.7.2) y Windows NT 4.0 con VC++ 5.0.

D.2 Fichero `copyrt.h`

Puede considerarse que todos los ficheros fuente siguientes tienen esta declaración de derechos de propiedad intelectual incorporada:

```
/*
** Copyright (c) 1990- 1993, 1996 Open Software Foundation, Inc.
** Copyright (c) 1989 by Hewlett-Packard Company, Palo Alto, Ca. &
** Digital Equipment Corporation, Maynard, Mass.
** Copyright (c) 1998 Microsoft.
** To anyone who acknowledges that this file is provided "AS IS"
** without any express or implied warranty: permission to use, copy,
** modify, and distribute this file for any purpose is hereby
** granted without fee, provided that the above copyright notices and
** this notice appears in all source code copies, and that none of
** the names of Open Software Foundation, Inc., Hewlett-Packard
** Company, or Digital Equipment Corporation be used in advertising
** or publicity pertaining to distribution of the software without
** specific, written prior permission. Neither Open Software
** Foundation, Inc., Hewlett-Packard Company, Microsoft, nor Digital
** Equipment Corporation makes any representations about the
** suitability
** of this software for any purpose.
*/
```

D.3 Fichero `uuid.h`

```
#include "copyrt.h"
#undef uuid_t
typedef struct {
    unsigned32    time_low;
    unsigned16    time_mid;
    unsigned16    time_hi_and_version;
    unsigned8     clock_seq_hi_and_reserved;
    unsigned8     clock_seq_low;
    byte          node[6];
} uuid_t;

/* uuid_create -- generate a UUID */
int uuid_create(uuid_t * uuid);

/* uuid_create_from_name -- create a UUID using a "name"
   from a "name space" */
void uuid_create_from_name(
    uuid_t *uuid,          /* resulting UUID */
    uuid_t nsid,           /* UUID of the namespace */
    void *name,            /* the name from which to generate a UUID */
    int namelen            /* the length of the name */
);

/* uuid_compare -- Compare two UUID's "lexically" and return
   -1  u1 is lexically before u2
   0   u1 is equal to u2
   1   u1 is lexically after u2

   Note that lexical ordering is not temporal ordering!
*/
int uuid_compare(uuid_t *u1, uuid_t *u2);
```

D.4 Fichero uuid.c

```

#include "copyrt.h"
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "sysdep.h"
#ifdef _WINDOWS_
#include <arpa/inet.h>
#endif
#include "uuid.h"

/* various forward declarations */
static int read_state(unsigned16 *clockseq, uuid_time_t *timestamp,
    uuid_node_t *node);
static void write_state(unsigned16 clockseq, uuid_time_t timestamp,
    uuid_node_t node);
static void format_uuid_v1(uuid_t *uuid, unsigned16 clockseq,
    uuid_time_t timestamp, uuid_node_t node);
static void format_uuid_v3(uuid_t *uuid, unsigned char hash[16]);
static void get_current_time(uuid_time_t *timestamp);
static unsigned16 true_random(void);

/* uuid_create -- generator a UUID */
int uuid_create(uuid_t *uuid)
{
    uuid_time_t timestamp, last_time;
    unsigned16 clockseq;
    uuid_node_t node;
    uuid_node_t last_node;
    int f;

    /* acquire system-wide lock so we're alone */
    LOCK;

    /* get time, node identifier, saved state from non-volatile storage */
    get_current_time(&timestamp);
    get_ieee_node_identifier(&node);
    f = read_state(&clockseq, &last_time, &last_node);

    /* if no NV state, or if clock went backwards, or node identifier
       changed (e.g., new network card) change clockseq */
    if ((f || memcmp(&node, &last_node, sizeof node))
        clockseq = true_random();
    else if (timestamp < last_time)
        clockseq++;

    /* save the state for next time */
    write_state(clockseq, timestamp, node);

    UNLOCK;

    /* stuff fields into the UUID */
    format_uuid_v1(uuid, clockseq, timestamp, node);
    return 1;
}

/* format_uuid_v1 -- make a UUID from the timestamp, clockseq,
   and node identifier */
void format_uuid_v1(uuid_t *uuid, unsigned16 clock_seq,
    uuid_time_t timestamp, uuid_node_t node)
{
    /* Construct a version 1 uuid with the information we've gathered
       plus a few constants. */
    uuid->time_low = (unsigned long)(timestamp & 0xFFFFFFFF);
    uuid->time_mid = (unsigned short)((timestamp >> 32) & 0xFFFF);
    uuid->time_hi_and_version =
        (unsigned short)((timestamp >> 48) & 0x0FFF);
    uuid->time_hi_and_version |= (1 << 12);
    uuid->clock_seq_low = clock_seq & 0xFF;
    uuid->clock_seq_hi_and_reserved = (clock_seq & 0x3F00) >> 8;
    uuid->clock_seq_hi_and_reserved |= 0x80;
    memcpy(&uuid->node, &node, sizeof uuid->node);
}

```

```

/* data type for UUID generator persistent state */
typedef struct {
    uuid_time_t  ts;          /* saved timestamp */
    uuid_node_t  node;        /* saved node identifier */
    unsigned16   cs;          /* saved Clock Sequence */
} uuid_state;

static uuid_state st;

/* read_state -- read UUID generator state from non-volatile store */
int read_state(unsigned16 *clockseq, uuid_time_t *timestamp,
               uuid_node_t *node)
{
    static int initied = 0;
    FILE *fp;

    /* only need to read state once per boot */
    if (!initied) {
        fp = fopen("state", "rb");
        if (fp == NULL)
            return 0;
        fread(&st, sizeof st, 1, fp);
        fclose(fp);
        initied = 1;
    }
    *clockseq = st.cs;
    *timestamp = st.ts;
    *node = st.node;
    return 1;
}

/* write_state -- save UUID generator state back to non-volatile
storage */
void write_state(unsigned16 clockseq, uuid_time_t timestamp,
                 uuid_node_t node)
{
    static int initied = 0;
    static uuid_time_t next_save;
    FILE* fp;

    if (!initied) {
        next_save = timestamp;
        initied = 1;
    }

    /* always save state to volatile shared state */
    st.cs = clockseq;
    st.ts = timestamp;
    st.node = node;
    if (timestamp >= next_save) {
        fp = fopen("state", "wb");
        fwrite(&st, sizeof st, 1, fp);
        fclose(fp);
        /* schedule next save for 10 seconds from now */
        next_save = timestamp + (10 * 10 * 1000 * 1000);
    }
}

/* get-current_time -- get time as 60-bit 100ns ticks since UUID epoch.
Compensate for the fact that real clock resolution is
less than 100ns. */
void get_current_time(uuid_time_t *timestamp)
{
    static int initied = 0;
    static uuid_time_t time_last;
    static unsigned16 uuids_this_tick;
    uuid_time_t time_now;

    if (!initied) {
        get_system_time(&time_now);
        uuids_this_tick = UUIDS_PER_TICK;
        initied = 1;
    }

    for ( ; ; ) {
        get_system_time(&time_now);

        /* if clock reading changed since last UUID generated, */
        if (time_last != time_now) {

```

```

        /* reset count of uuids gen'd with this clock reading */
        uuids_this_tick = 0;
        time_last = time_now;
        break;
    }
    if (uuids_this_tick < UUIDS_PER_TICK) {
        uuids_this_tick++;
        break;
    }
    /* going too fast for our clock; spin */
}
/* add the count of uuids to low order bits of the clock reading */
*timestamp = time_now + uuids_this_tick;
}

/* true_random -- generate a crypto-quality random number.
**This sample doesn't do that.** */
static unsigned int true_random(void)
{
    static int initd = 0;
    unsigned int time_now;

    if (!initd) {
        get_system_time(&time_now);
        time_now = time_now / UUIDS_PER_TICK;
        srand((unsigned int)((time_now >> 32) ^ time_now) & 0xffffffff);
        initd = 1;
    }

    return rand();
}

/* uuid_create_from_name -- create a UUID using a "name" from a "name
space" */
void uuid_create_from_name(uuid_t *uuid, unsigned int nsid, void *name,
                           int namelen)
{
    MD5_CTX c;

    unsigned char hash[16];
    unsigned int net_nsid;

    /* put name space identifier in network byte order so it hashes the
       same no matter what endian machine we're on */
    net_nsid = nsid;
    htonl(net_nsid.time_low);
    htons(net_nsid.time_mid);
    htons(net_nsid.time_hi_and_version);

    MD5Init(&c);
    MD5Update(&c, &net_nsid, sizeof net_nsid);
    MD5Update(&c, name, namelen);
    MD5Final(hash, &c);

    /* the hash is in network byte order at this point */
    format_uuid_v3(uuid, hash);
}

/* format_uuid_v3 -- make a UUID from a (pseudo)random 128-bit number */
void format_uuid_v3(uuid_t *uuid, unsigned char hash[16])
{
    /* convert UUID to local byte order */
    memcpy(uuid, hash, sizeof *uuid);
    ntohl(uuid->time_low);
    ntohs(uuid->time_mid);
    ntohs(uuid->time_hi_and_version);

    /* put in the variant and version bits */
    uuid->time_hi_and_version &= 0x0FFF;
    uuid->time_hi_and_version |= (3 << 12);
    uuid->clock_seq_hi_and_reserved &= 0x3F;
    uuid->clock_seq_hi_and_reserved |= 0x80;
}

/* uuid_compare -- Compare two UUID's "lexically" and return */
#define CHECK(f1, f2) if (f1 != f2) return f1 < f2 ? -1 : 1;
int uuid_compare(uuid_t *u1, uuid_t *u2)
{
    int i;

```

```

CHECK(u1->time_low, u2->time_low);
CHECK(u1->time_mid, u2->time_mid);
CHECK(u1->time_hi_and_version, u2->time_hi_and_version);
CHECK(u1->clock_seq_hi_and_reserved, u2->clock_seq_hi_and_reserved);
CHECK(u1->clock_seq_low, u2->clock_seq_low)
for (i = 0; i < 6; i++) {
    if (u1->node[i] < u2->node[i])
        return -1;
    if (u1->node[i] > u2->node[i])
        return 1;
}
return 0;
}
#undef CHECK

```

D.5 Fichero sysdep.h

```

#include "copyrt.h"
/* remove the following define if you aren't running Windows 32 */
#define WININC 0

#ifdef WININC
#include <windows.h>
#else
#include <time.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/time.h>
#endif

#include "global.h"
/* change to point to where MD5 .h's live; IETF RFC 1321 has a sample
   implementation */
#include "md5.h"

/* set the following to the number of 100ns ticks of the actual
   resolution of your system's clock */
#define UUIDS_PER_TICK 1024

/* set the following to a call to get and release a global lock */
#define LOCK
#define UNLOCK

typedef unsigned long    unsigned32;
typedef unsigned short   unsigned16;
typedef unsigned char     unsigned8;
typedef unsigned char     byte;

/* set this to what your compiler uses for 64-bit data type */
#ifdef WININC
#define unsigned64_t unsigned __int64
#define I64(C) C
#else
#define unsigned64_t unsigned long long
#define I64(C) C##LL
#endif

typedef unsigned64_t uuid_time_t;
typedef struct {
    char nodeID[6];
} uuid_node_t;

void get_ieee_node_identifier(uuid_node_t *node);
void get_system_time(uuid_time_t *uuid_time);
void get_random_info(unsigned char seed[16]);

```

D.6 Fichero sysdep.c

```

#include "copyrt.h"
#include <stdio.h>
#include <string.h>
#include "sysdep.h"

/* system dependent call to get MAC node identifier.
   This sample implementation generates a random node identifier. */
void get_ieee_node_identifier(uuid_node_t *node)
{

```

```

static int initd = 0;
static uuid_node_t saved_node;
unsigned char seed[16];
FILE *fp;

if (!initd) {
    fp = fopen("nodeid", "rb");
    if (fp) {
        fread(&saved_node, sizeof saved_node, 1, fp);
        fclose(fp);
    }
    else {
        get_random_info(seed);
        seed[0] |= 0x80;
        memcpy(&saved_node, seed, sizeof saved_node);
        fp = fopen("nodeid", "wb");
        if (fp) {
            fwrite(&saved_node, sizeof saved_node, 1, fp);
            fclose(fp);
        }
    }
    initd = 1;
}

*node = saved_node;
}
/* system dependent call to get the current system time. Returned as
   100ns ticks since UUID epoch, but resolution may be less than 100ns. */
#ifdef _WINDOWS_

void get_system_time(uuid_time_t *uuid_time)
{
    ULARGE_INTEGER time;

    /* Windows NT keeps time in FILETIME format which is 100ns ticks since
       Jan 1, 1601. UUIDs use time in 100ns ticks since Oct 15, 1582.
       The difference is 17 Days in Oct + 30 (Nov) + 31 (Dec)
       + 18 years and 5 leap days. */
    GetSystemTimeAsFileTime((FILETIME *)&time);
    time.QuadPart +=
        (unsigned __int64) (1000*1000*10) // seconds
        * (unsigned __int64) (60 * 60 * 24) // days
        * (unsigned __int64) (17+30+31+365*18+5); // # of days
    *uuid_time = time.QuadPart;
}

void get_random_info(unsigned char seed[16])
{
    MD5_CTX c;
    struct {
        MEMORYSTATUS m;
        SYSTEM_INFO s;
        FILETIME t;
        LARGE_INTEGER pc;
        DWORD tc;
        DWORD l;
        char hostname[MAX_COMPUTERNAME_LENGTH + 1];
    } r;

    MD5Init(&c);
    GlobalMemoryStatus(&r.m);
    GetSystemInfo(&r.s);
    GetSystemTimeAsFileTime(&r.t);
    QueryPerformanceCounter(&r.pc);
    r.tc = GetTickCount();
    r.l = MAX_COMPUTERNAME_LENGTH + 1;
    GetComputerName(r.hostname, &r.l);
    MD5Update(&c, &r, sizeof r);
    MD5Final(seed, &c);
}

#else

void get_system_time(uuid_time_t *uuid_time)
{
    struct timeval tp;

    gettimeofday(&tp, (struct timezone *)0);

```

```

/* Offset between UUID formatted times and Unix formatted times.
   UUID UTC base time is October 15, 1582.
   Unix base time is January 1, 1970. */
*uuid_time = (tp.tv_sec * 1000000) + (tp.tv_usec * 10)
             + I64(0x01B21DD213814000);
}

void get_random_info(unsigned char seed[16])
{
    MD5_CTX c;
    struct {
        struct timeval t;
        char hostname[257];
    } r;

    MD5Init(&c);
    gettimeofday(&r.t, (struct timezone *)0);
    gethostname(r.hostname, 256);
    MD5Update(&c, &r, sizeof r);
    MD5Final(seed, &c);
}

#endif

```

D.7 Fichero utable.c

```

#include "copyrt.h"
#include "sysdep.h"
#include <stdio.h>
#include "uuid.h"

uuid_t NameSpace_DNS = { /* 6ba7b810-9dad-11d1-80b4-00c04fd430c8 */
    0x6ba7b810,
    0x9dad,
    0x11d1,
    0x80, 0xb4, 0x00, 0xc0, 0x4f, 0xd4, 0x30, 0xc8
};

/* puid -- print a UUID */
void puid(uuid_t u)
{
    int i;
    printf("%8.8x-%4.4x-%4.4x-%2.2x%2.2x-", u.time_low, u.time_mid,
        u.time_hi_and_version, u.clock_seq_hi_and_reserved,
        u.clock_seq_low);
    for (i = 0; i < 6; i++)
        printf("%2.2x", u.node[i]);
    printf("\n");
}

/* simple driver for UUID generator */
int main(int argc, char **argv)
{
    uuid_t u;
    int f;

    uuid_create(&u);
    printf("uuid_create(): "); puid(u);

    f = uuid_compare(&u, &u);
    printf("uuid_compare(u,u): %d\n", f); /* should be 0 */
    f = uuid_compare(&u, &NameSpace_DNS);
    printf("uuid_compare(u, NameSpace_DNS): %d\n", f); /* should be 1 */
    f = uuid_compare(&NameSpace_DNS, &u);
    printf("uuid_compare(NameSpace_DNS, u): %d\n", f); /* should be -1 */
    uuid_create_from_name(&u, NameSpace_DNS, "www.widgets.com", 15);
    printf("uuid_create_from_name(): "); puid(u);
}

```

D.8 Ejemplo de salida de utable

```

uuid_create(): 7d444840-9dc0-11d1-b245-5ffdce74fad2
uuid_compare(u,u): 0
uuid_compare(u, NameSpace_DNS): 1
uuid_compare(NameSpace_DNS, u): -1
uuid_create_from_name(): e902893a-9d22-3c7e-a7b8-d6e313b71d9f

```

D.9 Algunos ID de espacio de nombre

En esta cláusula se enumeran los ID de espacio de nombre para algunos espacios de nombre potencialmente interesantes, como estructuras inicializadas en lenguaje C y en representación de la cadena que se de.

```

/* Name string is a fully-qualified domain name */
uuid_t Namespace_DNS = { /* 6ba7b810-9dad-11d1-80b4-00c04fd430c8 */
    0x6ba7b810,
    0x9dad,
    0x11d1,
    0x80, 0xb4, 0x00, 0xc0, 0x4f, 0xd4, 0x30, 0xc8
};

/* Name string is a URL */
uuid_t Namespace_URL = { /* 6ba7b811-9dad-11d1-80b4-00c04fd430c8 */
    0x6ba7b811,
    0x9dad,
    0x11d1,
    0x80, 0xb4, 0x00, 0xc0, 0x4f, 0xd4, 0x30, 0xc8
};

/* Name string is an OID */
uuid_t Namespace_OID = { /* 6ba7b812-9dad-11d1-80b4-00c04fd430c8 */
    0x6ba7b812,
    0x9dad,
    0x11d1,
    0x80, 0xb4, 0x00, 0xc0, 0x4f, 0xd4, 0x30, 0xc8
};

/* Name string is a Directory distinguished name (in DER or a text output format) */
uuid_t Namespace_X500 = { /* 6ba7b814-9dad-11d1-80b4-00c04fd430c8 */
    0x6ba7b814,
    0x9dad,
    0x11d1,
    0x80, 0xb4, 0x00, 0xc0, 0x4f, 0xd4, 0x30, 0xc8
};

```


BIBLIOGRAFÍA

- [1] ZAHN (L.), DINEEN (T.), LEACH (P.): Network Computing Architecture, *ISBN 0-13-611674-4*, January 1990.
- [2] Open Group CAE: DCE: Remote Procedure Call, *Specification C309, ISBN 1-85912-041-5*, August 1994.
- [3] ISO/CEI 11578:1996, *Information technology – Open Systems Interconnection – Remote Procedure Call (RPC)*.
- [4] IEEE, Request Form for an Individual Address Block (also known as an Ethernet Address Block) of 4,096 MAC Addresses, <http://standards.ieee.org/regauth/oui/pilot-ind.html>
- [5] Recomendación UIT-T X.500 (2001) | ISO/CEI 9594-1:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Visión de conjunto de conceptos, modelos y servicios*.
- [6] IETF RFC 3061 (2001), *A URN Namespace of Object Identifiers*.
- [7] Internet-Draft draft-mealling-uuid-urn-00, *A UUID URN Namespace*, M. Mealling, P. Leach, R. Salz, October 2002.

SERIES DE RECOMENDACIONES DEL UIT-T

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



Impreso en Suiza

Ginebra, 2005