Unión Internacional de Telecomunicaciones

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

X.1206

(04/2008)

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

Seguridad en el ciberespacio – Ciberseguridad

Marco independiente del proveedor para la notificación automática de información relacionada con la seguridad y para la difusión automática de actualizaciones

Recomendación UIT-T X.1206

Unión Internacional de Telecomunicaciones

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS, COMUNICACIONES DE SISTEMAS ABIERTOS Y SEGURIDAD

REDES PÚBLICAS DE DATOS	X.1–X.199
INTERCONEXIÓN DE SISTEMAS ABIERTOS	X.200–X.299
INTERFUNCIONAMIENTO ENTRE REDES	X.300–X.399
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	X.600–X.699
GESTIÓN DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	X.700–X.799
SEGURIDAD	X.800–X.849
APLICACIONES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	X.850–X.899
PROCESAMIENTO DISTRIBUIDO ABIERTO	X.900–X.999
SEGURIDAD DE LA INFORMACIÓN Y DE LAS REDES	
Aspectos generales de la seguridad	X.1000–X.1029
Seguridad de las redes	X.1030–X.1049
Gestión de la seguridad	X.1050–X.1069
Telebiometría	X.1080–X.1099
APLICACIONES Y SERVICIOS CON SEGURIDAD	
Seguridad en la multidifusión	X.1100–X.1109
Seguridad en la red residencial	X.1110–X.1119
Seguridad en las redes móviles	X.1120–X.1139
Seguridad en la web	X.1140–X.1149
Protocolos de seguridad	X.1150–X.1159
Seguridad en las comunicaciones punto a punto	X.1160–X.1169
Seguridad de la identidad en las redes	X.1170–X.1179
Seguridad en la TVIP	X.1180–X.1199
SEGURIDAD EN EL CIBERESPACIO	
Ciberseguridad	**X.1200–X.1229**
Lucha contra el correo basura	X.1230–X.1249
Gestión de identidades	X.1250–X.1279
APLICACIONES Y SERVICIOS CON SEGURIDAD	
Comunicaciones de emergencia	X.1300–X.1309
Seguridad en las redes de sensores ubicuos	X.1310–X.1339

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

Recomendación UIT-T X.1206

Marco independiente del proveedor para la notificación automática de información relacionada con la seguridad y para la difusión automática de actualizaciones

Resumen

En la presente Recomendación UIT-T X.1206 se describe un marco para la notificación automática de información relacionada con la seguridad y para la difusión automática de actualizaciones. La característica más importante de este marco es que no depende del proveedor. Una vez registrado el recurso, pueden ponerse automáticamente a la disposición de los usuarios, o directamente de las aplicaciones correspondientes, información actualizada sobre sus vulnerabilidades, y los parches ("*patches*") o actualizaciones del mismo.

Orígenes

La Recomendación UIT-T X.1206 fue aprobada el 18 de abril de 2008 por la Comisión de Estudio 17 (2005-2008) del UIT-T por el procedimiento de la Resolución 1 de la AMNT.

PREFACIO

La Unión Internacional de Telecomunicaciones (UIT) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones y de las tecnologías de la información y la comunicación. El Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T) 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 en la dirección http://www.itu.int/ITU-T/ipr/.

ÍNDICE

Recomendación UIT-T X.1206

Marco independiente del proveedor para la notificación automática de información relacionada con la seguridad y para la difusión automática de actualizaciones

1 Alcance

En la presente Recomendación se describe un marco para el flujo bidireccional de notificación y difusión automáticas de la información sobre las vulnerabilidades, así como la distribución de o actualizaciones o parches ("*patches*"). Además, permite que los administradores de sistemas conozcan el estado de cualquier recurso ("*asset*") que pertenezca a su ámbito de responsabilidad.

En las cláusulas 6 y 7 se describen los problemas de mantenimiento de los recursos desde el punto de vista de su identificación, la difusión de información y la gestión de sistemas/redes.

En la cláusula 8 figura una descripción general del marco independiente del proveedor, y un ejemplo de sistema basado en dicho marco, su funcionamiento y un ejemplo de una secuencia de intercambio de mensajes. También se tratan en esta cláusula los aspectos de seguridad que se han de tener en cuenta en este marco independiente del proveedor.

En la cláusula 9 se describen las funcionalidades y las características de la presente Recomendación.

La cláusula 10 proporciona las definiciones de las estructuras de datos de los componentes de la presente Recomendación.

La cláusula 11 contiene el código XML, definido y descrito en la cláusula 10.

En esta Recomendación se describe un marco al que pueden recurrir los proveedores para la notificación y la recepción de información sobre vulnerabilidades, y para la distribución de parches/actualizaciones relacionados con los recursos que pertenecen a su ámbito de responsabilidad. Se define además el formato de la información que ha de emplearse dentro de los componentes que se utilizan en este marco y entre ellos.

Esta Recomendación no define protocolos para la comunicación entre componentes, puesto que pueden utilizarse muchos sin mayor problema.

Si bien es necesario fijar algunas funciones y responsabilidades comunes para poder recurrir al marco independiente del proveedor, dichas funciones y las responsabilidades que de ellas emanan quedan fuera del alcance de esta Recomendación.

2 Referencias

Ninguna.

3 Definiciones

3.1 Términos definidos en esta Recomendación

En la presente Recomendación se definen los siguientes términos:

3.1.1 agente: Entidad que aplica esta Recomendación a un recurso instalado en un determinado dispositivo, que puede funcionar como servidor o como servidor local.

3.1.2 recurso (*asset*): Dispositivo, pieza de equipo independientemente identificable, aplicación, sistema operativo o código ejecutable.

3.1.3 cliente: Dispositivo que solicita servicios de otro dispositivo.

3.1.4 dispositivo: Sistema que actúa como cliente, servidor, o ambos, o bien como servidor local.

3.1.5 grupo: Conjunto de dispositivos que funcionan como una sola unidad.

3.1.6 servidor local: Cliente que actúa como nodo servidor para otros clientes en sentido descendente.

3.1.7 mensaje: Solicitud de una determinada acción; puede tratarse de una acción general, por ejemplo "Registrar" que un recurso dado es de una cierta versión y/o contiene componentes de determinadas versiones, "Solicitar" actualizaciones, parches o información sobre vulnerabilidades existentes o futuros, etc. En otros contextos ajenos a la funcionalidad de la presente Recomendación pueden definirse otros tipos de mensajes.

3.1.8 datos del mensaje: Información que se proporciona en un determinado mensaje. Entre las casi infinitas posibilidades que existen, pueden citarse algunos ejemplos concretos de datos definidos en esta Recomendación: información sobre la versión, información sobre las vulnerabilidades de determinadas versiones y parches o actualizaciones de versiones específicas.

3.1.9 conjunto de mensaje (*message set*): Combinación y asociación de un identificador único universal, un mensaje y los datos contenidos en el mismo, que se definen mediante un código XML generado a partir del *Message_Core* definido en esta Recomendación y que lo amplían.

3.1.10 parche: Modificación de difusión general que sirve para solucionar una vulnerabilidad específica de un producto relacionada con la seguridad. Método de actualización de un fichero que consiste en reemplazar únicamente las partes que se modifican, en lugar de todo el fichero.

3.1.11 servidor: Dispositivo que recibe las solicitudes de otros dispositivos.

3.1.12 vulnerabilidad: Cualquier punto débil, proceso o mecanismo administrativo, o exposición física de un computador o una red que sean susceptibles de ser aprovechados por una amenaza.

4 Siglas y acrónimos

En esta Recomendación se emplean las siguientes siglas y acrónimos:

API Interfaz de programación de aplicaciones (*application programming interface)*

GUID Identificador único global (*globally unique IDentifier*)

HTTP Protocolo de transferencia de hipertexto (*hypertext transfer protocol*)

ISIRT Equipos de intervención en caso de emergencia informática (*information security incident response team*)

ISP Proveedor de servicio Internet (*Internet service provider*)

OS Sistema operativo (*operating system*)

POAS Plataforma/sistema operativo/aplicación/servicio (*platform/operating system/application/service*)

URI Identificador de recursos uniforme (*uniform resource identifier*)

5 Convenciones

Ninguna.

6 Introducción

A medida que aumenta el número de personas que comienzan a utilizar computadores en sus trabajos y hogares, la mayoría sin la formación necesaria para ello, ni mucho menos en los aspectos relacionados con la seguridad, se acerca rápidamente el momento en que será imposible no solamente mantener un mínimo de seguridad sino que cada vez será más difícil para los encargados de la seguridad a nivel del sistema tener una idea del estado de los sistemas de los que se encargan y a los que prestan servicios, antes de que ocurra algún incidente o brecha en la seguridad, cuando ya sea demasiado tarde.

Esto se debe sobre todo a la gran cantidad de computadores diferentes que existe, todos en estados diferentes de mantenimiento y actualización. En lo que respecta a la seguridad, la gestión de sistemas es más un proceso de gestión de desastres y recuperación que uno de prevención.

Si bien varias aplicaciones, e incluso algunos sistemas operativos (OS), tienen sus propios mecanismos de actualización, todos tienen problemas en común, entre los que cabe citar que el mecanismo de actualización debe, en primer lugar, haber sido activado y, en segundo lugar, que sólo se ejecuta previa notificación al usuario de que existe una actualización disponible, siempre y cuando éste permita las notificaciones.

Probablemente la peor consecuencia de todos estos problemas sea que el administrador de sistema está completamente por fuera del proceso, de tal manera que si no instala su propio sistema de verificación en cada computador del que se encarga no puede tener idea del nivel general de seguridad en sus redes y sistemas.

Otro aspecto que cabe tener en cuenta es que si bien es importante actualizar el software con la versión más reciente disponible, a menudo las actualizaciones por sí solas no son la solución, sino que es necesario que el usuario mejore su forma de utilizar su sistema, para la cual no existe mejor "actualización" que si la información recibida por el usuario a este respecto es útil. Aunque algunas aplicaciones y sistemas operativos tienen sus propios mecanismos de actualización, ninguno de ellos dispone de una metodología uniforme para mantener informados a los usuarios acerca de las prácticas más recomendadas en aras de la seguridad.

Los métodos empleados para distribuir las actualizaciones también son importantes. Actualmente, todas las actualizaciones se reciben de las fuentes a través de canales especiales, uno por cada sesión de actualización. Sin embargo, si las actualizaciones, u otra información importante, se pusieran a disposición de varios centros de redistribución, por ejemplo, el ISP o las redes de la empresa, y luego se distribuyeran los paquetes dentro de cada red de una manera fiable y segura, se podría reducir a la mitad, o por lo menos reducir bastante, la anchura de banda necesaria para la distribución, tal como ocurre con los servidores intermedios (*proxies*) del protocolo de transferencia de hipertexto (HTTP).

Otro problema, que con frecuencia no se resuelve adecuadamente, se presenta cuando un usuario, que tiene dificultades respecto a la utilización de un recurso o de acciones relacionadas con él, no tiene a quien acudir. Aunque haya una manera de entrar en contacto con los administradores del sistema u otro personal de ayuda, a menudo el diálogo con ellos se limita a algo parecido al juego de las "20 preguntas", en el que el usuario y la persona que le ayuda se hacen preguntas mutuamente.

A menudo resulta difícil, incluso para quien tenga experiencia en el tema, conocer con exactitud los diversos recursos que componen el sistema correspondiente. Dado que las arquitecturas de software son modulares y que las diferentes piezas de software constan de módulos de diferentes fabricantes, cada uno con sus propios sistemas de gestión de versiones, incluso en el caso de que se disponga de una actualización o de una información pertinente sobre un determinado producto o un cierto módulo, es difícil qué saber a qué se debe aplicar y a quién, en particular para las personas sin conocimiento técnico, que desconocen los fundamentos de la seguridad de sus sistemas.

Al final, se llega a una situación en la que no se informa a los usuarios ni se tiene en cuenta a los encargados de la seguridad del sistema.

7 Situación actual con respecto a la información sobre vulnerabilidad

Actualmente, muchos proveedores y diversas organizaciones relacionadas con temas de seguridad, tal es el caso de los equipos de intervención en caso de emergencia informática (ISIRT, *information security incident response team*) informan acerca de vulnerabilidades y, cuando procede, suministran actualizaciones y parches. No obstante, a menudo los usuarios hacen caso omiso de dicha información, actualizaciones o parches, y ni siquiera saben si ello les concierne.

Pese a que esta situación se debe a varias razones, cabe comenzar por preguntarse por qué los usuarios tienen dificultades para utilizar la información que se les suministra.

Los usuarios se sirven de la información sobre la versión de los recursos para determinar si su sistema, software o componentes pueden verse afectados por una vulnerabilidad que se acaba de detectar. No obstante, al utilizar la información sobre la versión para detectar el sistema, el software o los componentes afectados puede presentar ciertas dificultades, en particular:

- Ambigüedad en la notación de la versión.

Las reglas de notación de la versión varían de un proveedor a otro, con lo cual es posible que los usuarios interpreten dicha información de una manera errónea. Si los usuarios no comprenden el sistema que utiliza el proveedor para la notación de la versión, tal vez no sepan que una determinada actualización, parche o información les concierne.

Por ejemplo, supongamos que hay 3 versiones, "v1", "v1.1"y "v1.2", del sistema destinatario de cierta información sobre vulnerabilidad; si dicha información se describe como sigue:

"Sistemas afectados

* XXXX v1 el sistema afectado"

Se puede interpretar de dos maneras:

1) v1 es solamente v1.0.

2) v1 indica v1.x, es decir todas las versiones subordinadas a la versión 1.

No es práctico tratar de normalizar esta notación, por lo que sigue siendo necesario leer y entender las reglas específicas que emplea cada proveedor, aunque no es realista esperar que los usuarios lo hagan.

- La versión de un determinado producto no siempre sirve como criterio para establecer su vulnerabilidad.

En el caso de algunos proveedores, es posible que la versión de los productos que se presenta a los usuarios no sirva para determinar si es vulnerable o no, (figura 1).

En la mayoría de los productos orientados a los componentes, la presencia de la vulnerabilidad debe establecerse sobre la base de la versión de cada componente, en lugar de la del producto.

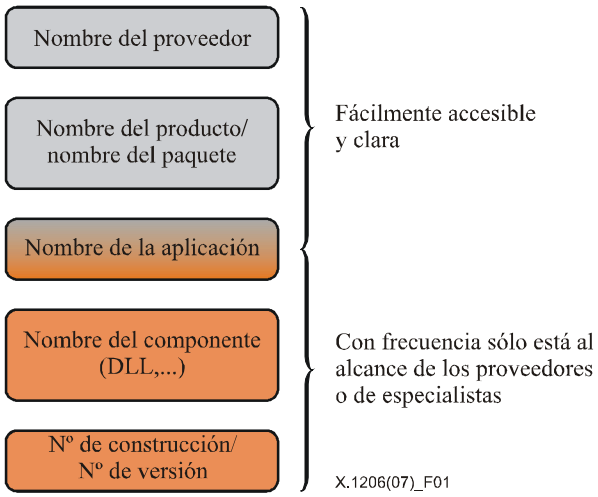


Figura 1 – Estructura jerárquica de la "versión"

- Productos incorporados como componentes en productos de otros proveedores

A menudo la información sobre vulnerabilidad se refiere a un producto determinado. Es posible que un usuario no se sirva de él directamente, sino que lo utilice a través de otra aplicación. De hecho, con las interfaces de programación de aplicaciones (API) abiertas es difícil que un proveedor sepa si sus productos están siendo utilizados en otras aplicaciones, o en cuáles están cargados o se utilizan.

- Los administradores de sistemas no conocen el estado de los recursos que pertenecen a su ámbito de responsabilidad.

Los administradores de sistemas dependen de que los usuarios hagan el mantenimiento a sus propios sistemas, y es imposible que conozcan el estado de los sistemas bajo su responsabilidad sin antes haber efectuado una verificación de cada uno de dichos sistemas o sin contar con la autonotificación. Sin esta última, que suele ser imprecisa e incompleta, por las muchas razones ya mencionadas que tienen que ver con las versiones, y sin aplicaciones de supervisión adaptadas, para las cuales se requiere desarrollo y mantenimiento, los encargados de la seguridad de las redes empresariales o las de los proveedores de servicio Internet (ISP) disponen de pocas herramientas para llevar a cabo esa tarea.

8 Descripción general del marco independiente del proveedor

En esta cláusula se suministra una descripción general del marco independiente del proveedor para la distribución de información sobre vulnerabilidad, actualizaciones y parches.

Si se adopta este marco, se pueden construir los sistemas de distribución de información sobre vulnerabilidad, actualizaciones y parches, como se muestra en la figura 2, utilizando un método simple de suscripción. Cada recurso, dispositivo o servidor local se puede inscribir con alguno o con todos los servidores disponibles para recibir información sobre vulnerabilidad, actualizaciones o parches, previa solicitud (*pull*) o por recomendación (*push*).

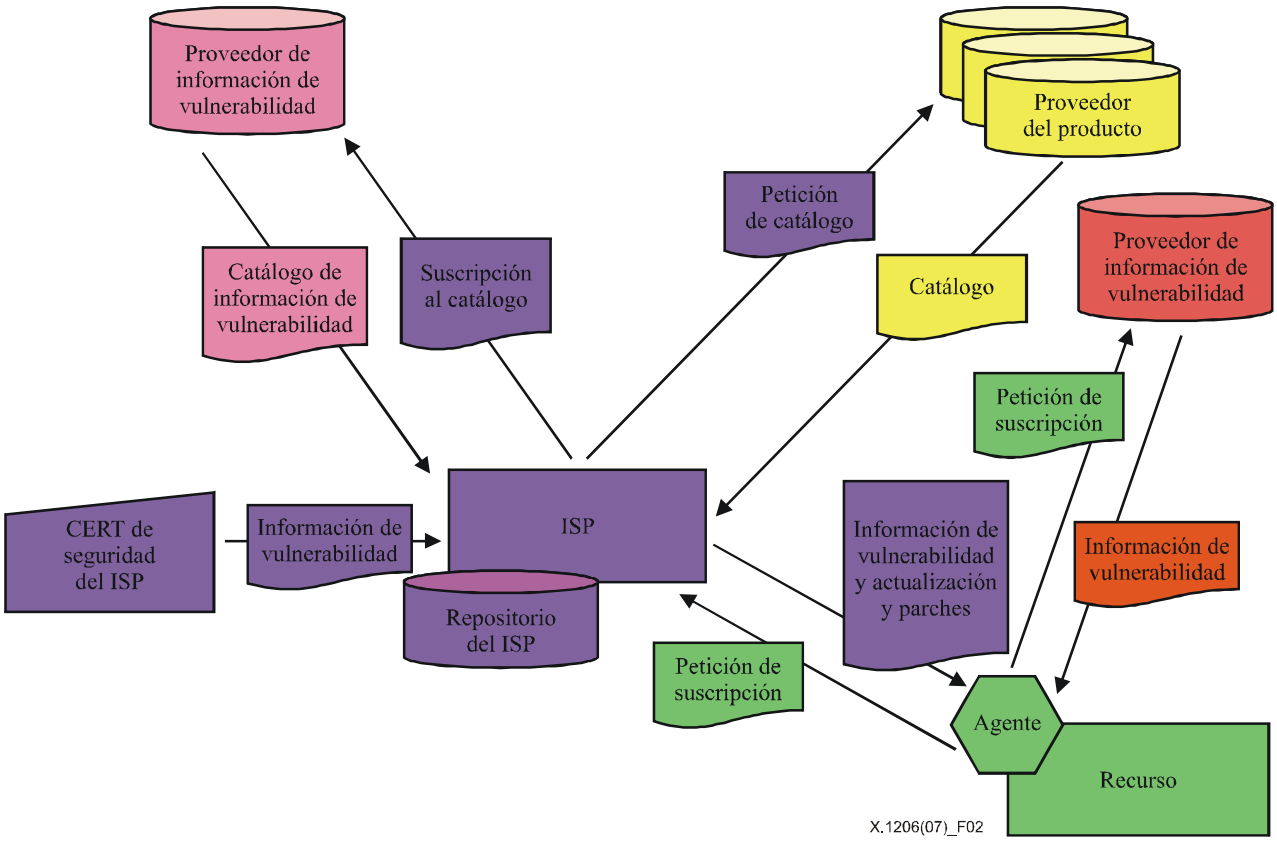


Figura 2 – Ejemplo de arquitectura de aplicación

8.1 Fuentes múltiples de información sobre vulnerabilidad, actualizaciones y parches

El método descrito permite que cualquier entidad solicite o proporcione a otra entidad información sobre vulnerabilidad, actualizaciones o parches. En el ejemplo de la figura, el ISP es la fuente desde la que se envía la información a todos sus abonados. Los abonados también se pueden registrar ante terceros independientes, a fin de que éstos les suministren información actualizada, análisis de códigos e incluso actualizaciones o parches para los códigos (programas). Para que una aplicación pueda efectuar una actualización o parche debe disponer de la autorización correspondiente, aunque en esta Recomendación no se indican los mecanismos para ello, salvo si se trata de una cierta utilización de las firmas.

8.2 Ejemplo del funcionamiento de una aplicación

8.2.1 Proceso de solicitud y suministro de información y actualizaciones

Se instala el recurso, por ejemplo, un editor de texto, y durante la instalación se notifica de ello a un agente instalado previamente. A partir de la información que recibe, el agente efectúa una solicitud de suscripción al ISP del usuario, con el fin de recibir toda la información sobre vulnerabilidad, las actualizaciones y/o los parches.

Antes de recibir la solicitud de suscripción del agente de usuario, o como resultado de ello, el ISP entra en contacto con varias fuentes de información sobre vulnerabilidad, actualizaciones o parches, basándose en la información proporcionada por el agente, por ejemplo, con la fuente o el proveedor de la aplicación recién instalada.

Del mismo modo, el agente del usuario puede, en este ejemplo, solicitar información sobre vulnerabilidad a otras fuentes y ponerla a disposición del usuario. Para ello se puede recurrir a un sitio web que preste servicios de fuente de información sobre vulnerabilidad y disponga de un enlace que permita descargar ficheros, o a un mensaje definido conforme a lo especificado en la presente Recomendación, que tras su recepción el agente puede actuar correspondientemente.

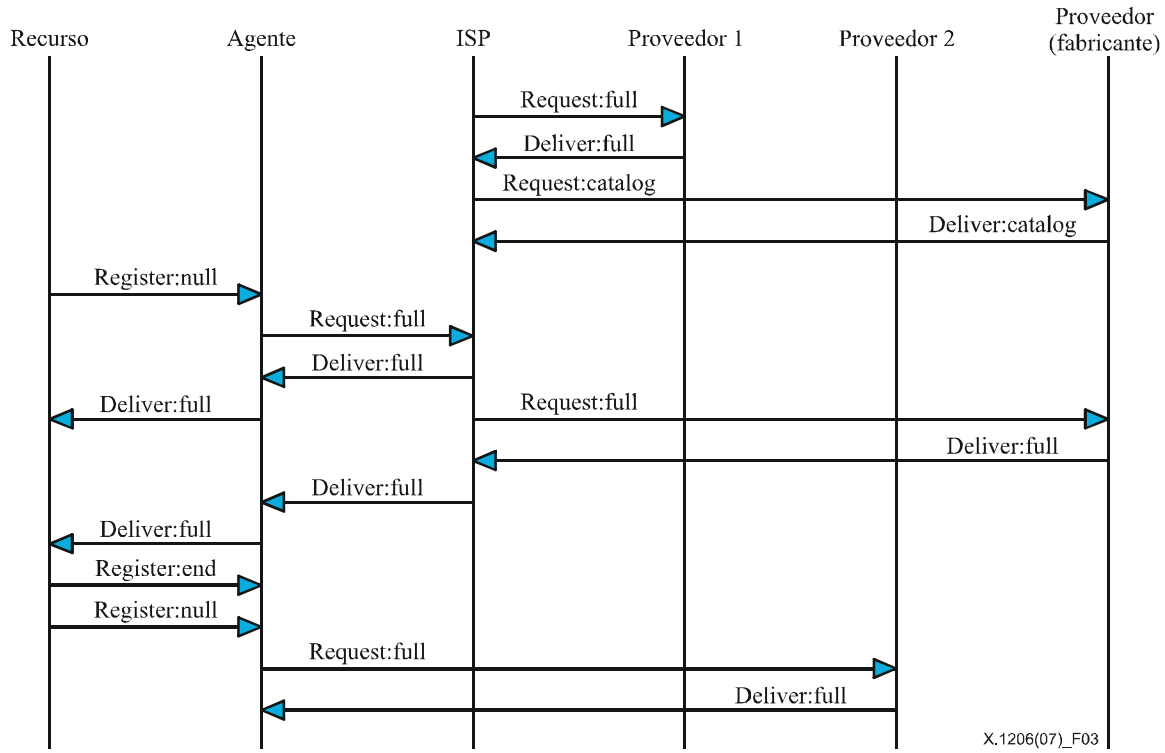


Figura 3 – Ejemplo de mensajes de aplicación

8.2.2 Mensaje de solicitud y suministro de información y actualizaciones

Un ISP, a menudo el administrador de una red empresarial o privada, solicita a varios proveedores información sobre vulnerabilidad, actualizaciones o parches. Puede solicitarse todo el contenido disponible, de tal manera que el ISP pueda fungir como un repositorio local desde el principio, o simplemente una lista (Catálogo) de la de información sobre vulnerabilidad, actualizaciones o parches disponibles, a partir de la cual se pueda pedir más adelante los elementos necesarios. Aunque no se incluya en la figura, puede ocurrir que una petición de todo el contenido reciba como respuesta "No se soporta el mensaje principal" ("*Major Message Not Supported*") o también "No se soporta el mensaje secundario" ("*Minor Message Not Supported*"), en cuyo caso el servicio que hizo la petición sabrá que en su lugar se debe solicitar una lista (Catálogo). No obstante, es preferible que los proveedores hagan públicas sus políticas en materia de solicitudes, con lo cual se evita negociarla.

En este ejemplo particular, si bien el ISP solicita únicamente un catálogo del proveedor, pues el repositorio completo de éste puede ser bastante voluminoso, puede estar seguro de que todo lo que recibe el agente, en nombre de sus usuarios, corresponde a la versión más reciente disponible, por cuanto una solicitud de catálogo también implica una suscripción a toda información futura. De otra parte, dado que es probable que el proveedor de información sobre vulnerabilidad, por ejemplo, el Proveedor 1, tenga en línea sólo una parte de la información, tal vez sea apropiado solicitar toda la información disponible.

Durante la instalación de un recurso, el proceso de instalación puede enviar un mensaje *Request:Full* (Tipos de mensaje principal: secundario) a un agente instalado localmente, en el que se solicite recibir toda la información sobre vulnerabilidad, actualizaciones y parches actuales y futuros, a medida que vayan siendo disponibles, para la versión específica del recurso o de cualquiera de sus componentes.

Luego, el agente envía un *Request:Full* al ISP en nombre del medio recientemente instalado.

Dado que en este caso el ISP ya posee en su repositorio información sobre vulnerabilidad del Proveedor 1, puede suministrarla instantáneamente o esperar hasta que se reciba una respuesta del proveedor (fabricante), para luego combinar ambas informaciones y enviarlas.

Habiendo establecido el ISP que el proveedor es una fuente de información válida para el recurso, a partir del catálogo que tiene almacenado, verifica entonces en su memoria *caché* si la información está disponible localmente o, si no fuere le caso, le solicita toda la información disponible y la pasa al agente, pudiendo también almacenarla en su propio repositorio.

A través de una interfaz de usuario con el agente, el usuario puede encontrar por sí mismo fuentes valiosas de información sobre vulnerabilidad y, utilizando tal vez algún formulario de solicitud predeterminado de un proveedor, por ejemplo, el Proveedor 2, que el usuario haya telecargado, encargar al agente que solicite un catálogo o la información sobre vulnerabilidad disponible, o a medida que vaya apareciendo.

Tras haber actualizado un determinado recurso o componente, se cancelan las solicitudes de actualización de las versiones anteriores mediante el mensaje *Register:End,* y se solicitan las actualizaciones del caso para las nuevas versiones a través de un mensaje *Register:null*. Igualmente, si se saca de servicio un recurso, también es posible cancelar las actualizaciones futuras de información valiéndose de un *Register:End*.

8.3 Aspectos relacionados con la seguridad

En el marco de esta Recomendación, no se transmite al proveedor (fabricante) información sobre el usuario. Sin embargo, es necesario tener en cuenta la autenticación y la verificación de la integridad de la información que se entrega, para lo cual se sugiere utilizar los siguientes métodos:

- Las fuentes deben identificarse mediante firmas en los mensajes.
- Los agentes deben verificar la autenticidad y la integridad de los mensajes.
- Las actualizaciones no deben estar personalizadas, para que la información que se transmita por la red no sea información específica del sistema de usuario.
- La descarga o instalación en el equipo de usuario requiere la aprobación previa de éste.

9 Arquitectura recomendada

9.1 Capa de núcleo del mensaje

El núcleo del mensaje (*message core*) permite una comunicación independiente de la plataforma/el sistema operativo/la aplicación/el servicio (POAS, *platform/operating system/application/service*) entre el servidor conforme y las aplicaciones cliente que presentan requisitos dependientes del POAS. La capa del núcleo del mensaje compendia, para las aplicaciones conformes a un determinado mensaje, la complejidad inherente a ciertas operaciones, con lo cual el administrador de sistema sólo debe ocuparse de las operaciones que han de realizarse.

9.2 Capa de mensaje/aplicación

Capa que consta de dos elementos principales. El primero son las funciones y los procesos que son independientes del POAS, que se llevan a cabo durante las operaciones normales del recurso o del dispositivo, por ejemplo, la solicitud de información sobre vulnerabilidad, actualizaciones o parches. Se trata de operaciones de alto nivel que se aplican a sistemas y funciones sin importar como hayan sido implementados. Por ejemplo, en todos los sistemas se requiere periódicamente que los usuarios estén al tanto de las actualizaciones, la aplicación de parches o la recepción de información sobre vulnerabilidad a medida que vayan siendo disponibles.

El problema reside en que prácticamente cada sistema lleva a cabo dichas funciones de alto nivel de una manera diferente al nivel de implementación. En la arquitectura definida en la presente Recomendación, es en esta capa donde los requisitos independientes del POAS entran en conflicto con implementaciones de nivel inferior, que se crea la interfaz primaria.

En general, se atribuye un identificador a los procesos/operaciones de alto nivel que han de ser ejecutados en el sistema, con independencia del tipo de sistema. Por consiguiente, los sistemas de tipos diferentes utilizan dichos identificadores para comunicarse instrucciones entre sí. Tras recibir una determinada instrucción, el sistema la hace pasar, junto con toda la información correspondiente, a un sistema específico encargado de procesar dicha instrucción.

En cierta manera, es similar al tratamiento y proyección de una secuencia de vídeo comprimido o a la lectura de un fichero, tal como el presente documento, en varias plataformas diferentes. Cada dispositivo de vídeo de una plataforma o cada lector de ficheros se encarga de procesar la información recibida basándose en ciertas normas preestablecidas.

9.3 Escalabilidad

Los sistemas que aplican la presente Recomendación son escalables gracias a la arquitectura topográfica de identificación de servidor/cliente, en la cual es posible que un cliente de un servidor sea el servidor de otros clientes de nivel inferior. De hecho, toda agrupación topográfica de servidores y clientes se puede describir mediante nodos en un diagrama de árbol. Asimismo, gracias a la capacidad de agrupar los mensajes, con arreglo a su pertenencia a un cierto cliente o grupo, cualquier servidor, en cualquier nivel por encima de un determinado cliente, puede comunicarse e interactuar con dicho cliente.

- Los clientes y los servidores se identifican mediante un identificador único global (GUID) de 128 bits que se les atribuye.
- Es posible atribuir previamente a los clientes un GUID, reatribuirles uno durante el registro con determinado servidor, o ambos.
- En función de la política en materia de topografía, es posible que a un cliente se atribuyan, o utilice, diferentes GUID para interactuar con diferentes servidores.
- Cuando un cliente sea a su vez servidor de otros clientes (servidor local), el servidor local puede "esconder" el verdadero GUID del servidor del que depende, o viceversa.
- Los clientes pueden depender o ser atribuidos a más de un servidor, cada uno de los cuales proporciona una gama diferente de servicios.

9.4 Extensibilidad

Se pueden añadir nuevas plataformas, aplicaciones, funciones y servicios creando simplemente un esquema que se derive del esquema de mensaje principal de esta Recomendación y lo extienda.

A partir de un esquema de mensaje definido, es posible entonces crear módulos para cualquier plataforma con la seguridad de que habrá compatibilidad total.

Basándose en los mensajes y las estructuras de datos de mensaje del nuevo esquema de mensajes, es posible crear con facilidad, u omitir, interfaces de usuario.

Los mensajes comunes se pueden "reutilizar" en nuevos conjuntos de mensajes, ya sea importando la definición de mensaje existente, la estructura de datos de mensajes, o ambas, según convenga.

9.5 Independencia de la plataforma

- Esta Recomendación es la interfaz "normativa" entre todas las entidades participantes.
- Los módulos que implementen la Recomendación se crean dependientes del POAS con el fin de utilizar mensajes, funciones y servicios, según corresponda.
- Todo dispositivo que participe puede enviar cualquier mensaje o petición especificados a cualquier otro dispositivo participante, sin importar en cuál plataforma están funcionando ambos dispositivos. En algunos casos particulares, los datos de mensaje contenidos pueden incluir contenido específico de la plataforma, aunque en todo caso la Recomendación fue elaborada para que haga innecesaria la necesidad de saber en qué plataforma funcionan las partes. La Recomendación sólo acepta que un dispositivo ordene a otro lo que debe hacer, aunque no cómo hacerlo.

9.6 Comunicación cliente/servidor

9.6.1 Protocolo público

- Se basa en XML.

9.6.2 Modos de comunicación servidor/cliente

- Impuesto por el servidor
  - El servidor envía al cliente las notificaciones y las actualizaciones para las que se ha registrado, a medida que van siendo.
- Solicitado por el servidor
  - El servidor envía al cliente solicitudes de operaciones, notificaciones y actualizaciones.
- Impuesto por el cliente
  - El cliente envía notificaciones y actualizaciones al servidor.
- Solicitado por el cliente
  - El cliente envía al servidor solicitudes de operaciones, notificaciones y actualizaciones.

10 Componentes del marco

La Recomendación se define utilizando XML y consta de una arquitectura de mensajería para el direccionamiento del origen y el destino de los mensajes, y el transporte de los mensajes y de los correspondientes datos.

En esta cláusula se proporcionan las definiciones del formato de datos de los componentes principales: el contenedor de mensaje, el registro de mensaje y la versión de mensaje.

10.1 Contenedor de mensaje

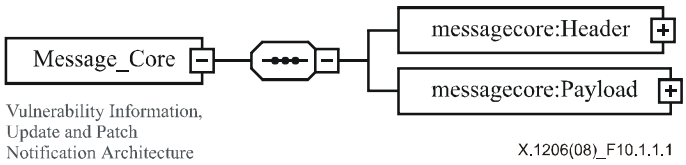
El contenedor de mensaje actúa como un empaque para el transporte, encaminamiento y entrega de mensajes y respuestas, y como una base abstracta mediante la cual se pueden crear, por extensión y derivación, otros mensajes y respuestas.

10.1.1 Message_Core

Todos los mensajes en que se emplea la arquitectura de esta Recomendación se definen mediante la importación, la extensión y la derivación del *Message_Core*.

El elemento raíz de la estructura de mensaje es el *Message_Core*.

10.1.1.1 Sintaxis



10.1.1.2 Semántica

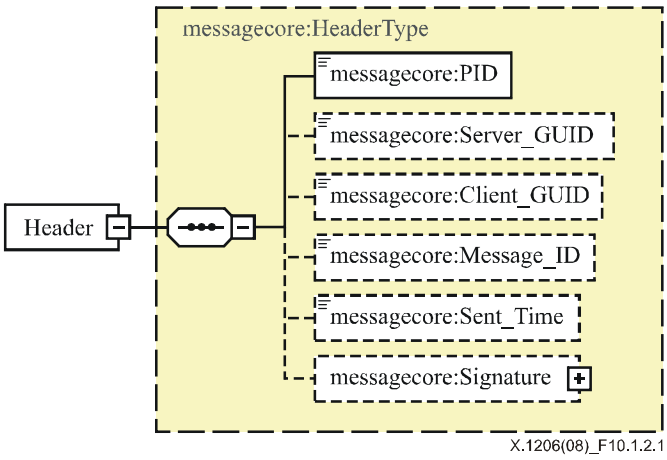
Header (*encabezamiento*) – Contiene información general de encaminamiento e identificación de mensaje. **[OBLIGATORIO]**

Payload (*cabida útil*) – Contiene una **selección (*choice*)** XML de uno o varios mensajes agrupados *Message_Core* o un solo mensaje. Se emplea en situaciones en las que los mensajes pueden almacenarse en caché en un determinado nodo (servidor local) o cuando un servidor que controle/preste servicio a un servidor local desee recibir mensajes o datos de mensajes de clientes controlados/o a los que presta servicio el servidor local. **[OBLIGATORIO]**

10.1.2 Encabezamiento

El encabezamiento de mensaje contiene un ID de protocolo, con fines de extensibilidad y compatibilidad, información de direccionamiento, tales como la o las fuentes y el o los destinos de un determinado mensaje, un campo que lleva un ID de mensaje y un campo que contiene la información de la hora de origen del mensaje.

10.1.2.1 Sintaxis



10.1.2.2 Semántica

PID – ID de protocolo. Un valor de 128 bits que identifica una versión específica, una extensión al protocolo de mensaje o un reemplazo completo, que lleva como mínimo el *Message_Core*, el *Header*, y el *PID*. Para efectuar la señalización con arreglo a la presente Recomendación, las aplicaciones pueden utilizar el valor #h00000001. **[OBLIGATORIO]**

Server_GUID – Un valor de 128 bits del que se sirven los clientes para identificar unívocamente un servidor o un servidor local del cual dependen directamente. **[FACULTATIVO]**

Client_GUID – Un valor de 128 bits que utilizan los servidores o los servidores locales para identificar unívocamente clientes que dependen de ellos. **[FACULTATIVO]**

Message_ID – Un valor de 128 bits que sirve para identificar unívocamente un determinado mensaje. Si bien el método empleado en la selección de un *Message_ID* adecuado no es normativo, cabe esperar que un sistema homólogo determinado utilice el mismo valor de *Message_ID* en las respuestas subsiguientes. **[FACULTATIVO]**

Sent_Time – Contiene la hora a la que se envió un determinado mensaje. **[FACULTATIVO]**

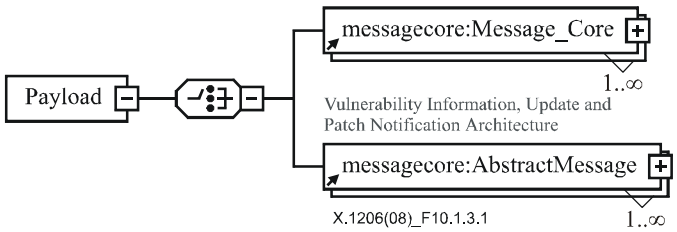
Signature – Una firma en XML. **[FACULTATIVO]**

10.1.3 Cabida útil

La funcionalidad correspondiente al contenido de una cierta cabida útil (*Payload)* se define mediante el establecimiento y la asociación de un *UUID* y/o un *CID* con un *Message* y los *Message_Data* correspondientes. Los elementos *UUID/CID*, *Message* y *Message_Data* son el punto principal de extensibilidad en la arquitectura de la Recomendación y son los que le confieren su independencia de la plataforma. Un *Message* definido y sus *Message_Data* correspondientes son lo que se conocen como un conjunto de mensaje (*Message Set*).

Gracias a la definición y asociación de un *UUID* y/o un *CID* con la definición de un conjunto de mensaje en esquema XML, que importa y extiende el esquema *Message_Core*, y a la implementación de un módulo para un determinado conjunto de mensaje, que incluye el esquema *Message* y *Message_Data*, se pueden utilizar módulos en cualquier aplicación conforme a esta Recomendación, con total compatibilidad.

10.1.3.1 Sintaxis



10.1.3.2 Semántica

Message_Core – Proporciona la capacidad de empaquetar y entregar varios mensajes de diferentes fuentes. Si se escoge es **[OBLIGATORIO]**

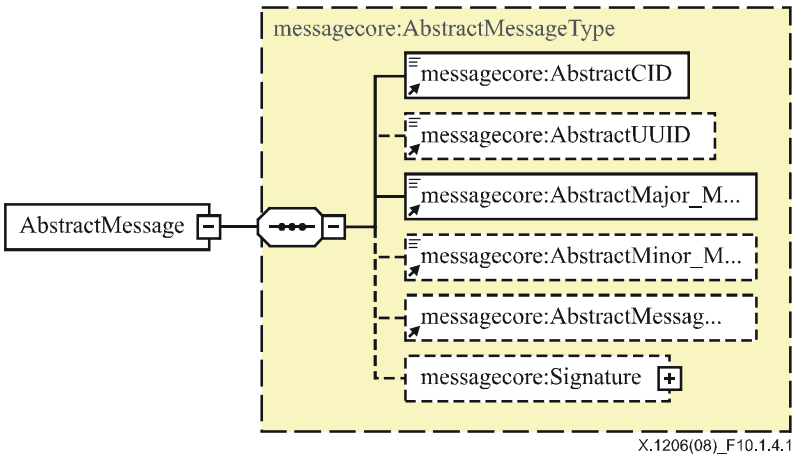
AbstractMessage – El identificador pertinente del mensaje y del módulo, así como la indicación del tipo específico de mensaje y de los datos específicos de mensaje. **[OBLIGATORIO]**

10.1.4 AbstractMessage

La estructura de datos AbstractMessage sirve para transportar información que se emplea no sólo para identificar exactamente el mensaje y el modo de mensaje que se están solicitando, sino también para llevar toda la información necesaria para el transporte del mensaje solicitado.

Cualquier mensaje específico que se defina a través de un esquema normativo puede extender esta estructura de datos, razón por la cual, al implementarse en el núcleo como una clase abstracta, esta estructura no puede emplearse sin una definición que la extienda.

10.1.4.1 Sintaxis



10.1.4.2 Semántica

AbstractCID – Un valor de 128 bits que sirve para identificar unívocamente una sola funcionalidad determinada. Un determinado módulo implementado, identificado mediante un *UUID*, puede aceptar más de un tipo de funcionalidad identificada por un *CID*. Por su parte, más de un módulo puede implementar la misma funcionalidad, aunque para diferentes aplicaciones, por ejemplo, un módulo de una versión de un procesador de texto y un módulo de una versión de un editor de vídeo. **[OBLIGATORIO]**

AbstractUUID – Un valor de 128 bits utilizado para identificar unívocamente un módulo implementado que puede procesar los *Messages* correspondientes a su ID. **[FACULTATIVO]**

AbstractMajor_Message – Una cadena que se emplea para transportar instrucciones desde los servidores a los clientes o viceversa. Dichas instrucciones se definen en esquemas que importan y extienden el esquema *Message_Core* y que pueden incluir, aunque no están limitados a, "Register" (registrar), "Request" (solicitar), etc. **[OBLIGATORIO]**

AbstractMinor_Message – Una cadena adicional de identificador de mensaje que sirve para definir el modo del mensaje indicado. Dichas instrucciones se definen en esquemas que importan y extienden el esquema *Message_Core* y los modos pueden ser, aunque no se limitan a, "Full" (completo), "Catalog" (catálogo), etc. **[FACULTATIVO]**

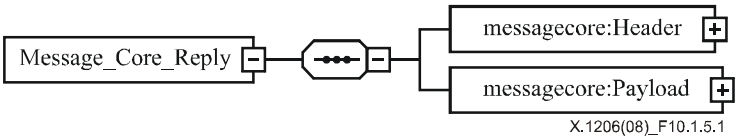
AbstractMessage_Data – Datos necesarios para el tratamiento del mensaje solicitado. **[FACULTATIVO]**

Signature – Una firma en XML. **[FACULTATIVO]**

10.1.5 Message_Core_Reply

La estructura de datos *Message_Core_Reply* sirve para transportar una respuesta de estado y cualquier otro dato deseado hacia cualquier otra entidad que haya iniciado un determinado tren de comunicaciones.

10.1.5.1 Sintaxis



10.1.5.2 Semántica

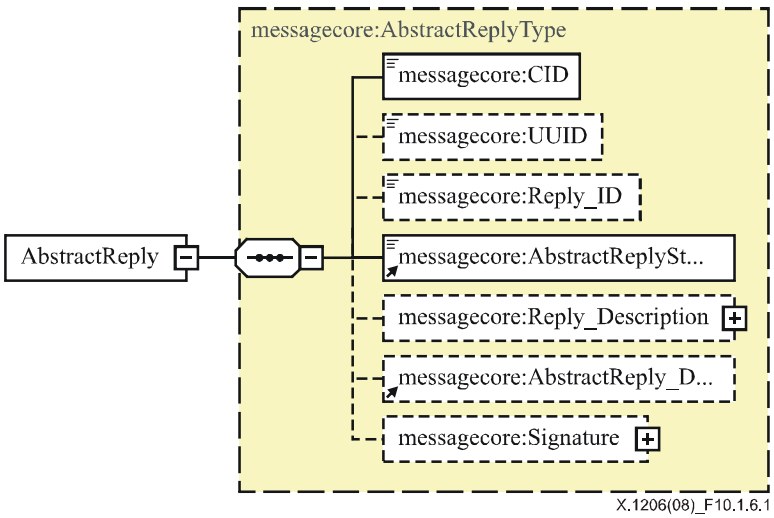
Header – Estructura, datos y requisitos idénticos a los del *Message_Core:Header*.

Payload – Contiene una selección XML de uno o varios mensajes agrupados *Message_Core_Reply* o un solo mensaje *Message_Core_Reply*. Se utiliza cuando es posible almacenar los mensajes en memoria caché en un determinado nodo (servidor local) o cuando un servidor que controla/presta servicio a un servidor local desea recibir mensajes y datos de mensaje de clientes controlados por o a los que presta servicios el servidor local. **[OBLIGATORIO]**

10.1.6 Abstract_Reply

La estructura de datos *Abstract_Reply* sirve para transportar una respuesta de estado y cualquier otro dato deseado hacia cualquier otra entidad que haya iniciado un determinado tren de comunicaciones.

10.1.6.1 Sintaxis



10.1.6.2 Semántica

CID – Valor de 128 bits que permite identificar una versión y/o una extensión específicas del protocolo de mensaje, o un reemplazo completo que importa como mínimo *Message_Core*, *Header*, y *PID*. Para efectuar la señalización con arreglo a la presente Recomendación, las aplicaciones pueden utilizar el valor #h00000001. **[OBLIGATORIO]**

UUID – Un valor de 128 bits utilizado para identificar unívocamente un módulo implementado que puede procesar las respuestas correspondientes a su ID. **[FACULTATIVO]**

Reply_ID – Un valor de 128 bits que sirve para identificar unívocamente una determinada respuesta. Si bien el método empleado para seleccionar una *Reply_ID* adecuada no es normativo, se espera que un sistema par utilice el mismo *Message_ID* en las peticiones a las cuales se envía esta respuesta. **[FACULTATIVO]**

AbstractReplyString – Cadena extendida por los mensajes que importan el esquema *Message_Core* y que contiene cadenas que indican el estado de la petición anterior a la cual se responde. Puede tratarse, aunque no se limita a, "Failed" (fallo), "Succeeded" (éxito), "Message not supported" (mensaje no válido), etc. **[OBLIGATORIO]**

Reply_Description – Texto o HTML con el que se proporciona más información acerca del *AbstractReplyString*. **[FACULTATIVO]**

AbstractReply_Data – Toda información requerida que depende de la implementación. **[FACULTATIVO]**

Signature – Una firma XML. **[FACULTATIVO]**

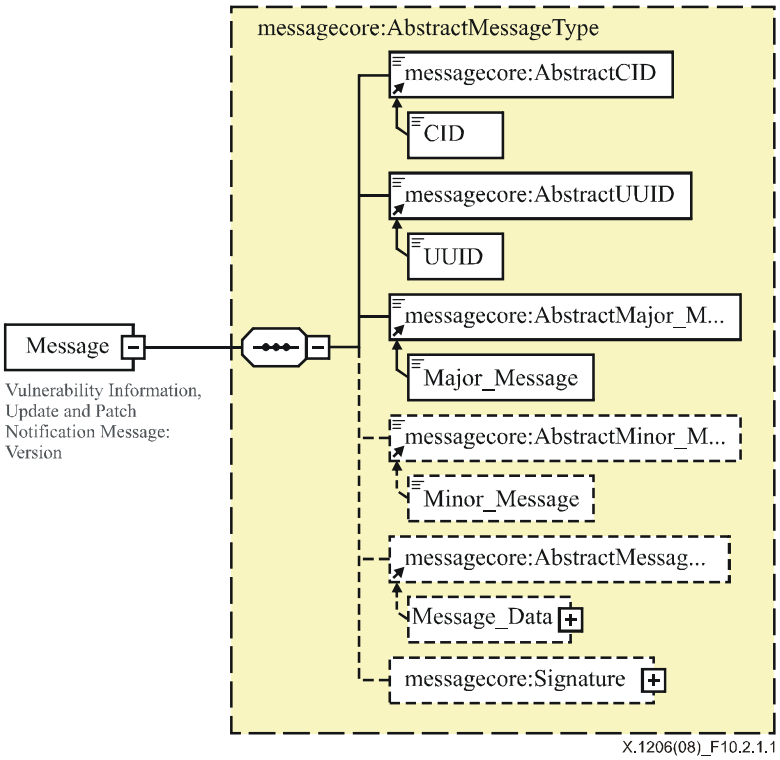
10.2 Mensaje de versión

Lo utilizan los componentes de un sistema para solicitarse y ponerse a disposición entre sí información sobre vulnerabilidad, actualizaciones o parches.

10.2.1 Version:Message

Version:Message es un substituto directo allí donde exista *AbstractMessage*, y su estructura de datos se restringe y extiende de tal manera que acepte únicamente el proceso que consiste en poner a disposición, solicitar y entregar información sobre vulnerabilidad, actualizaciones o parches.

10.2.1.1 Sintaxis



10.2.1.2 Semántica

CID – Reemplaza y restringe la clase de base *Message_Core:AbstractCID* por un valor de #h02 específicamente atribuido. Identifica unívocamente este mensaje como uno que pertenece a la clase de mensaje versión. **[OBLIGATORIO]**

UUID – Reemplaza la clase de base Message *Message_Core:AbstractUUID* y sirve para identificar un módulo implementado que puede procesar *Messages* con su ID correspondiente. **[OBLIGATORIO]**

Major_Message – Reemplaza y restringe la clase de base *Message_Core:AbstractMajor_Message* por una lista enumerada de valores posibles "Register", "Request", "Deliver" (entregar), "Analyse" (analizar). **[OBLIGATORIO]**

Los valores se utilizan de la siguiente manera:

- "Register" (registrar) – Notifica al cliente que recibe que el recurso y su versión, conforme a la descripción de *Version:Message_Data*, están siendo utilizados y que se debe entregar la información sobre vulnerabilidad, las actualizaciones o los parches a media que vayan siendo disponibles.

- "Request" (solicitar) – Solicita la información sobre vulnerabilidad, las actualizaciones o los parches disponibles para los productos que satisfagan, como mínimo, los requisitos especificados en *Version:Message_Data*.
- "Deliver" (suministrar) – Se incluyen en *Version:Message_Data* información sobre vulnerabilidad, y una o varias actualizaciones o parches.
- "Analyse" (analizar) – Se solicita que se analice la información que contiene *Version:Message_Data*. Se puede saber *a priori* el porqué de dicha solicitud o se puede ofrecer una explicación en *version:Info – version:Description o version:Info – version:Container – version:Description*, en función de lo que mejor convenga a la aplicación.

Minor_Message – Reemplaza y restringe la clase de base *Message_Core:AbstractMinor_Message* por una lista enumerada de posibles valores "Full", "Catalog" y "End". **[FACULTATIVO]**

Los valores se utilizan de la siguiente manera:

- "Full" (todo) – Solicita toda la información sobre vulnerabilidad, las actualizaciones o los parches disponibles.
- "Catalog" (catálogo) – Entrega una lista de la información sobre vulnerabilidad, las actualizaciones o los parches disponibles para los productos especificados en *Version:Message_Data*.
- "End" (fin) – Termina una suscripción con arreglo a la información contenida en *Version:Message_Data*.

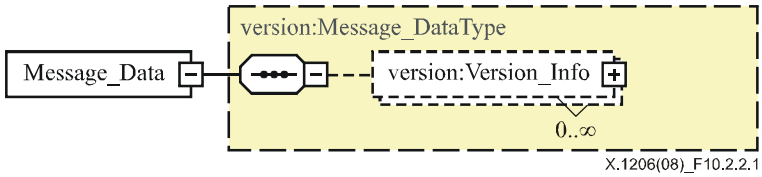
Message_Data – Reemplaza y restringe la clase de base *Message_Core:AbstractMessage_Data* por una estructura de datos específica que sirve para transportar la identificación de la versión de producto, y también lleva toda la información sobre vulnerabilidad, las actualizaciones o los parches aplicables. **[FACULTATIVO]**

Signature – Una firma XML. **[FACULTATIVO]**

10.2.2 Version:Message_Data

Version:Message_Data es un substituto directo allí donde exista *AbstractMessage_Data*, y su estructura de datos se restringe y extiende de tal manera que se acepte únicamente el proceso que consiste en poner a disposición, solicitar y entregar información sobre vulnerabilidad, actualizaciones o parches.

10.2.2.1 Sintaxis



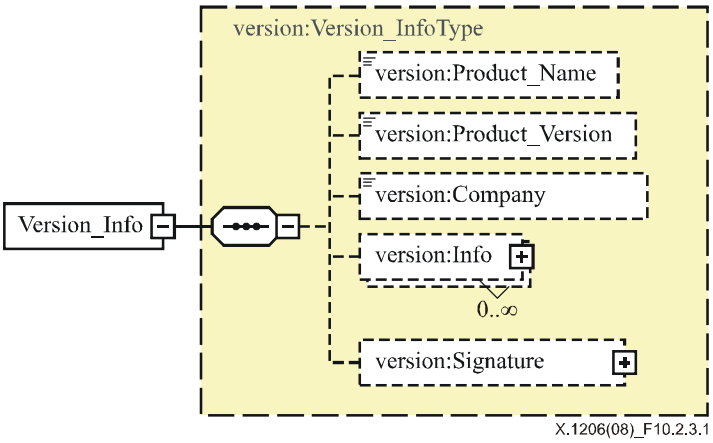
10.2.2.2 Semántica

Version_Info – Información que se emplea para identificar, hasta el nivel requerido de especificidad, los productos y sus versiones, así como el soporte de información sobre vulnerabilidad, actualizaciones o parches.

10.2.3 Version_Info

Sirve no solamente para identificar productos, junto con su proveedor, su versión y la información relacionada con la versión, sino también para transportar información sobre vulnerabilidad, actualizaciones o parches, según lo requiera la distribución a las aplicaciones que lo soliciten.

10.2.3.1 Sintaxis



10.2.3.2 Semántica

Product_Name – El nombre del producto de nivel superior al que se refiere la versión identificada, por ejemplo, en el caso de un determinado lector de DVD que contiene una versión específica del sistema *flash*, el *Product_Name* se referirá al nombre del producto lector DVD. **[FACULTATIVO]**

Product_Version – Versión del producto de nivel superior al que se refiere la información de versión identificada. En el ejemplo anterior, la *Product_Version* respondería a la versión específica del lector DVD. **[FACULTATIVO]**

Company (*empresa*) – El fabricante, proveedor o autor del producto o recurso identificado. **[FACULTATIVO]**

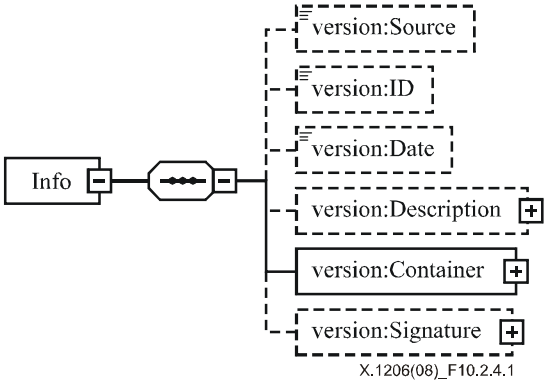
Info – Estructura de datos que sirve para transportar información sobre vulnerabilidad para implementaciones de productos o recursos, conforme a lo indicado por los valores de los campos mencionados. **[FACULTATIVO]**

Signature – Una firma XML. **[FACULTATIVO]**

NOTA – La información contenida en Info debe emplearse siempre para identificar un determinado recurso con el *Product_Name* y la *Product_Version* que se están empleando, sólo por conveniencia, aunque tratándose de un producto de nivel superior, conviene utilizar *Product_Name* y *Product_Version* para indicar claramente que no existe "progenitor" para el recurso identificado.

10.2.4 Info

10.2.4.1 Sintaxis



10.2.4.2 Semántica

Source (*origen*) – Descripción textual, por ejemplo, el nombre, del vendedor o del proveedor de información sobre vulnerabilidad que proporciona el código incluido o la información sobre vulnerabilidad. **[FACULTATIVO]**

ID – ID mediante el cual se puede identificar un determinado conjunto de datos *Info*. Aunque no se especifica un método de atribución, se sugiere que los ID atribuidos sólo han de ser únicos en una serie relacionada con un cierto recurso que proviene de una *Source* dada. **[FACULTATIVO]**

Date (*fecha*) – Valor de fecha XML correspondiente a *Info*. **[FACULTATIVO]**

Description (*descripción*) – Descripción textual o HTML del código contenido en un ejemplar facultativo de *Info_Container*, o de una referencia de identificador de recursos uniforme (URI, *uniform resource identifier*), para la versión del recurso descrita en un ejemplar específico de esta estructura de datos. **[FACULTATIVO]**

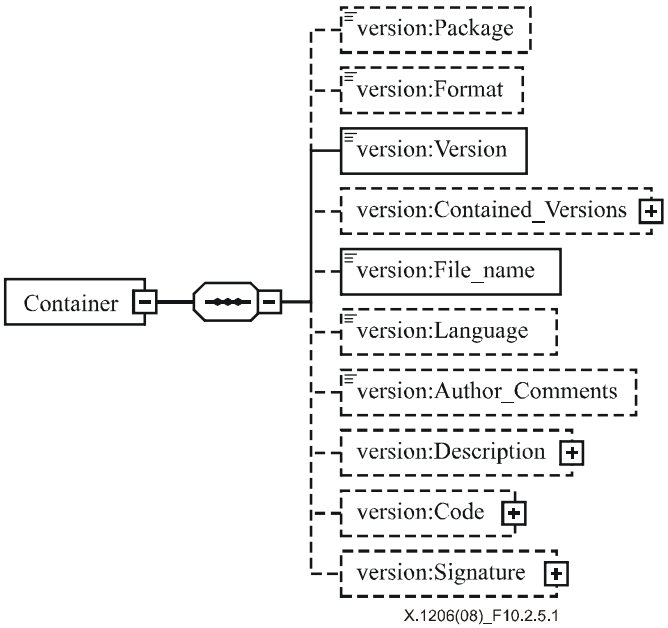
Container (*Contenedor*) – Estructura de datos utilizada para transportar código en forma de actualización o parche. **[OBLIGATORIO]**

Signature – Firma XML. **[FACULTATIVO]**

10.2.5 Contenedor

El que contiene la identificación de la versión, al igual que información sobre vulnerabilidad, una actualización o un parche.

10.2.5.1 Sintaxis



10.2.5.2 Semántica

Package (*paquete*) – Método de empaquetamiento utilizado por la versión específica y/o el del *Code*. **[FACULTATIVO]**

Format (*formato*) – Formato de la versión específica. **[FACULTATIVO]**

Version (*versión*) – La versión, bien sea que se incluya para las peticiones o se utilice para identificar el contenido de *Version_Control* y/o *Code*. **[OBLIGATORIO]**

Contained_Versions – Lista que se define mediante la estructura de datos *Version_Info* (véase *supra*) y que contiene descripciones de todos los componentes del recurso o del componente correspondientes. **[FACULTATIVO]**

File_Name – Nombre del fichero. **[FACULTATIVO]**

Language (*idioma*) – Tratándose de recursos para los cuales se dispone de varias versiones para los diversos países, teniendo en cuenta el idioma, este campo sirve para identificar el idioma. **[FACULTATIVO]**

Author_Comments – Cualquier texto que se quiera incluir. **[FACULTATIVO]**

Description (*descripción*) – Descripción textual o HTML de la versión específica contenida en el *Code*, o una descripción textual o HTML de la información sobre vulnerabilidad relacionada con el recurso identificado mediante *Version* y/o *File_Name* utilizados por o en el marco de *Version_Info : Product_Name* y *Version_Info : Product_Version*.

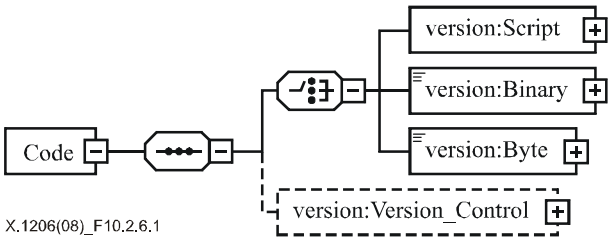
Code (*código*) – Contenedor en el que se almacena código ejecutable, al igual que información de control de versión.

Signature – Firma XML. **[FACULTATIVO]**

10.2.6 Código

Contenedor en el que se almacena código o cadenas de bytes, tal como una memoria, o una referencia URL, así como información útil para el control de las actualizaciones de la versión.

10.2.6.1 Sintaxis



10.2.6.2 Semántica

Script – Código en forma Script. **[RECOMENDADO]**

Binary (*binario*) – Código compilado en formato binario. **[RECOMENDADO]**

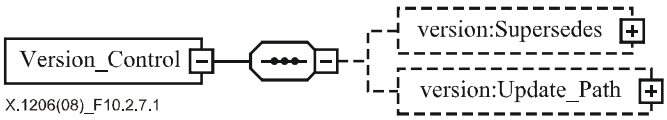
Byte – Datos proporcionados para el análisis. **[RECOMENDADO]**

Version_Control – Estructura de datos que permite indicar relaciones entre versiones sucesivas y sus posibles caminos de actualización. **[FACULTATIVO]**

10.2.7 Version_Control

Contenedor que transporta listas de secuencias de versiones reemplazadas por una determinada versión, y una lista que identifica a cuáles versiones se puede aplicar una actualización o un parche.

10.2.7.1 Sintaxis



10.2.7.2 Semántica

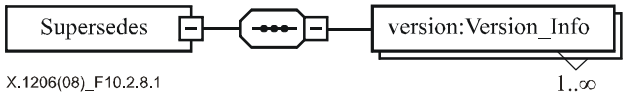
Supersedes (*reemplazadas*) – Lista secuencial de las versiones reemplazadas por la *Version_Info*. **[FACULTATIVO]**

Update_Path – Lista de versiones a las que se puede aplicar la actualización o el parche en cuestión. **[FACULTATIVO]**

10.2.8 Reemplazados

Lista de los recursos que han sido reemplazados por la versión específica.

10.2.8.1 Sintaxis



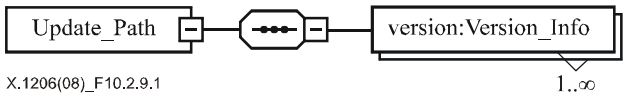
10.2.8.2 Semántica

Version_Info – (*Véase info*).

10.2.9 Update_Path

Lista de recursos a los que se puede aplicar la actualización o el parche en cuestión.

10.2.9.1 Sintaxis



X.1206(08)_F10.2.9.1

10.2.9.2 Semántica

Version_Info – (*Véase info*).

10.2.10 ReplyStatus

10.2.10.1 Sintaxis

X.1206(08)_F10.2.10.1

10.2.10.2 Semántica

ReplyStatus – Contiene uno de los siguientes:

"Failed" – No se pudo completar la operación solicitada.

"Succeeded" – La operación solicitada tuvo éxito.

"Major Message Not Supported" – No se soporta el identificador de mensaje principal solicitado ni su utilización en el contexto dado.

"Minor Message Not Supported" – No se soporta el identificador de mensaje secundario solicitado ni su utilización en el contexto dado.

"Handler not available" – No se soporta el tipo de mensaje identificado.

11 Esquemas

11.1 Message_Core

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:messagecore="http://www.itu.int//xml-namespace/itu-t/x.1206/CORE/" xmlns:ns1="http://www.w3.org/2000/09/xmldsig#" targetNamespace=" http://www.itu.int//xml-namespace/itu-t/x.1206/CORE/" elementFormDefault="qualified" attributeFormDefault="unqualified">

    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" schemaLocation="xmldsig-core-schema.xsd"/>

    <xs:element name="Message_Core">

        <xs:annotation>

            <xs:documentation>Vulnerability Information, Update and Patch Notification Architecture</xs:documentation>

        </xs:annotation>

        <xs:complexType>

            <xs:sequence>

                <xs:element name="Header" type="messagecore:HeaderType"/>

                <xs:element name="Payload">

                    <xs:complexType>

                        <xs:choice>

                            <xs:element ref="messagecore:Message_Core" maxOccurs="unbounded"/>

                            <xs:element ref="messagecore:AbstractMessage" maxOccurs="unbounded"/>

                        </xs:choice>

                    </xs:complexType>

                </xs:element>
```

```
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="HeaderType">
        <xs:sequence>
            <xs:element name="PID" type="xs:string"/>
            <xs:element name="Server_GUID" type="xs:string" minOccurs="0"/>
            <xs:element name="Client_GUID" type="xs:string" minOccurs="0"/>
            <xs:element name="Message_ID" type="xs:string" minOccurs="0"/>
            <xs:element name="Sent_Time" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="Signature" type="ns1:SignatureType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="AbstractMessage" type="messagecore:AbstractMessageType" abstract="true"/>
    <xs:complexType name="AbstractMessageType">
        <xs:sequence>
            <xs:element ref="messagecore:AbstractCID"/>
            <xs:element ref="messagecore:AbstractUUID" minOccurs="0"/>
            <xs:element ref="messagecore:AbstractMajor_Message"/>
            <xs:element ref="messagecore:AbstractMinor_Message" minOccurs="0"/>
            <xs:element ref="messagecore:AbstractMessage_Data" minOccurs="0"/>
            <xs:element name="Signature" type="ns1:SignatureType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="AbstractCID" type="messagecore:AbstractCIDType" abstract="true"/>
    <xs:complexType name="AbstractCIDType">
        <xs:simpleContent>
            <xs:extension base="xs:string"/>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="AbstractUUID" type="messagecore:AbstractUUIDType" abstract="true"/>
    <xs:complexType name="AbstractUUIDType">
        <xs:simpleContent>
            <xs:extension base="xs:string"/>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="AbstractMajor_Message" type="messagecore:AbstractMajorMessageType" abstract="true"/>
    <xs:complexType name="AbstractMajorMessageType">
        <xs:simpleContent>
            <xs:extension base="xs:string"/>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="AbstractMinor_Message" type="messagecore:AbstractMinorMessageType" abstract="true"/>
    <xs:complexType name="AbstractMinorMessageType">
        <xs:simpleContent>
            <xs:extension base="xs:string"/>
        </xs:simpleContent>
    </xs:complexType>
```

```
    <xs:element name="AbstractMessage_Data" type="messagecore:AbstractMessage_DataType" abstract="true"/>
    <xs:complexType name="AbstractMessage_DataType"/>
    <xs:element name="Message_Core_Reply">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Header" type="messagecore:HeaderType"/>
                <xs:element name="Payload">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element ref="messagecore:Message_Core_Reply" maxOccurs="unbounded"/>
                            <xs:element ref="messagecore:AbstractReply"/>
                        </xs:choice>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="AbstractReply" type="messagecore:AbstractReplyType"/>
    <xs:complexType name="AbstractReplyType">
        <xs:sequence>
            <xs:element name="CID" type="messagecore:AbstractCIDType"/>
            <xs:element name="UUID" type="messagecore:AbstractUUIDType" minOccurs="0"/>
            <xs:element name="Reply_ID" type="xs:string" minOccurs="0"/>
            <xs:element ref="messagecore:AbstractReplyStatus"/>
            <xs:element name="Reply_Description" type="messagecore:Description_Type" minOccurs="0"/>
            <xs:element ref="messagecore:AbstractReply_Data" minOccurs="0"/>
            <xs:element name="Signature" type="ns1:SignatureType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="AbstractReplyStatus" type="messagecore:AbstractReplyStatusType" abstract="true"/>
    <xs:complexType name="AbstractReplyStatusType">
        <xs:simpleContent>
            <xs:extension base="xs:string"/>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="AbstractReply_Data" type="messagecore:AbstractReply_DataType" abstract="true"/>
    <xs:complexType name="AbstractReply_DataType"/>
    <xs:complexType name="Description_Type">
        <xs:attribute name="ref" type="xs:anyURI" use="optional"/><![CDATA[]]></xs:complexType>
</xs:schema>
```

11.2 Message_Version

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:messagecore=" http://www.itu.int//xml-namespace/itu-t/x.1206/CORE/" xmlns:version=" http://www.itu.int//xml-namespace/itu-t/x.1206/CORE/MESSAGE/VERSION/" xmlns:xmldsig="http://www.w3.org/2000/09/xmldsig#" targetNamespace=" http://www.itu.int//xml-namespace/itu-t/x.1206/CORE/MESSAGE/VERSION/" elementFormDefault="qualified" attributeFormDefault="unqualified">

    <xs:import namespace="http://www.example.com/CORE" schemaLocation="Message_Core.xsd"/>

    <xs:import namespace="http://www.w3.org/2000/09/xmldsig#" schemaLocation="xmldsig-core-schema.xsd"/>

    <xs:element name="Message" type="messagecore:AbstractMessageType" substitutionGroup="messagecore:AbstractMessage">
```

```
        <xs:annotation>
            <xs:documentation>Vulnerability Information, Update and Patch Notification Message : Version</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="CID" type="messagecore:AbstractCIDType" substitutionGroup="messagecore:AbstractCID"/>
    <xs:element name="UUID" type="messagecore:AbstractUUIDType" substitutionGroup="messagecore:AbstractUUID"/>
    <xs:element name="Major_Message" type="version:Major_MessageType" substitutionGroup="messagecore:AbstractMajor_Message"/>
    <xs:complexType name="Major_MessageType">
        <xs:simpleContent>
            <xs:restriction base="messagecore:AbstractMajorMessageType">
                <xs:enumeration value="Register"/>
                <xs:enumeration value="Request"/>
                <xs:enumeration value="Deliver"/>
                <xs:enumeration value="Analyse"/>
            </xs:restriction>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="Minor_Message" type="version:Minor_MessageType" substitutionGroup="messagecore:AbstractMinor_Message"/>
    <xs:complexType name="Minor_MessageType">
        <xs:simpleContent>
            <xs:restriction base="messagecore:AbstractMinorMessageType">
                <xs:enumeration value="Full"/>
                <xs:enumeration value="Catalog"/>
                <xs:enumeration value="End"/>
            </xs:restriction>
        </xs:simpleContent>
    </xs:complexType>
    <xs:element name="Message_Data" type="version:Message_DataType" substitutionGroup="messagecore:AbstractMessage_Data"/>
    <xs:complexType name="Message_DataType">
        <xs:complexContent>
            <xs:extension base="messagecore:AbstractMessage_DataType">
                <xs:sequence>
                    <xs:element name="Version_Info" type="version:Version_InfoType" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="Reply" type="messagecore:AbstractReplyType" substitutionGroup="messagecore:AbstractReply"/>
    <xs:element name="ReplyStatus" substitutionGroup="messagecore:AbstractReplyStatus">
        <xs:complexType>
            <xs:simpleContent>
                <xs:restriction base="messagecore:AbstractReplyStatusType">
                    <xs:enumeration value="Failed"/>
                    <xs:enumeration value="Succeeded"/>
                    <xs:enumeration value="Major Message Not Supported"/>
                    <xs:enumeration value="Minor Message Not Supported"/>
                    <xs:enumeration value="Handler not available"/>
                </xs:restriction>
```

```
        </xs:simpleContent>
    </xs:complexType>
</xs:element>
<xs:element name="Reply_Data" type="messagecore:AbstractReply_DataType" substitutionGroup="messagecore:AbstractReply_Data"/>
<xs:complexType name="Version_InfoType">
    <xs:complexContent>
        <xs:extension base="messagecore:AbstractReply_DataType">
            <xs:sequence>
                <xs:element name="Product_Name" type="xs:string" minOccurs="0"/>
                <xs:element name="Product_Version" type="xs:string" minOccurs="0"/>
                <xs:element name="Company" type="xs:string" minOccurs="0"/>
                <xs:element name="Info" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Source" type="xs:string" minOccurs="0"/>
                            <xs:element name="ID" type="xs:string" minOccurs="0"/>
                            <xs:element name="Date" type="xs:string" minOccurs="0"/>
                            <xs:element name="Description" minOccurs="0">
                                <xs:complexType>
                                    <xs:attribute name="ref" type="xs:anyURI" use="optional"/><![CDATA[]]></xs:complexType>
                            </xs:element>
                            <xs:element name="Container">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="Package" type="xs:string" minOccurs="0"/>
                                        <xs:element name="Format" type="xs:string" minOccurs="0"/>
                                        <xs:element name="Version" type="xs:string"/>
                                        <xs:element name="Contained_Versions" type="version:Version_InfoType" minOccurs="0"/>
                                        <xs:element name="File_name" type="xs:string" minOccurs="0"/>
                                        <xs:element name="Language" type="xs:string" minOccurs="0"/>
                                        <xs:element name="Author_Comments" type="xs:string" minOccurs="0"/>
                                        <xs:element name="Description" type="messagecore:Description_Type" minOccurs="0"/>
                                        <xs:element name="Code" minOccurs="0">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:choice>
                                                        <xs:element name="Script">
                                                            <xs:complexType>
                                                                <xs:attribute name="ref" type="xs:anyURI" use="optional"/>
                                                            </xs:complexType>
                                                        </xs:element>
                                                        <xs:element name="Binary">
                                                            <xs:complexType>
                                                                <xs:simpleContent>
                                                                    <xs:extension base="xs:base64Binary">
                                                                        <xs:attribute name="ref" type="xs:anyURI" use="optional"/>
                                                                    </xs:extension>
                                                                </xs:simpleContent>
```

```
                                        </xs:complexType>
                                    </xs:element>
                                    <xs:element name="Byte">
                                        <xs:complexType>
                                            <xs:simpleContent>
                                                <xs:extension base="xs:base64Binary">
                                                    <xs:attribute name="ref" type="xs:anyURI" use="optional"/>
                                                </xs:extension>
                                            </xs:simpleContent>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:choice>
                                <xs:element name="Version_Control" minOccurs="0">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="Supersedes" minOccurs="0">
                                                <xs:complexType>
                                                    <xs:sequence>
                                                        <xs:element name="Version_info"
type="version:Version_InfoType" maxOccurs="unbounded"/>
                                                    </xs:sequence>
                                                </xs:complexType>
                                            </xs:element>
                                            <xs:element name="Update_Path" minOccurs="0">
                                                <xs:complexType>
                                                    <xs:sequence>
                                                        <xs:element name="Version_Info"
type="version:Version_InfoType" maxOccurs="unbounded"/>
                                                    </xs:sequence>
                                                </xs:complexType>
                                            </xs:element>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="Signature" type="xmldsig:SignatureType" minOccurs="0"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="Signature" type="xmldsig:SignatureType" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
```

```
                    <xs:element name="Signature" type="xmldsig:SignatureType" minOccurs="0"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

Bibliografía

[b-ITU-T X.667] Recomendación UIT-T X.667 (2004) | ISO/CEI 9834-8:2005, *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*.

[b-IETF RFC 3023] IETF RFC 3023 (2001), *XML Media Types*. <http://www.ietf.org/rfc/rfc3023.txt?number=3023>

[b-IETF RFC 3075] IETF RFC 3075 (2001), *XML-Signature Syntax and Processing*. <http://www.ietf.org/rfc/rfc3075.txt?number=3075>

[b-XML Datatypes] W3C Datatypes:2001, *XML Schema Part 2: Datatypes*, W3C Recommendation, Copyright © [2 May 2001] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). <http://www.w3.org/TR/2001/RECxmlschema-2-20010502/>

[b-XML Signature] W3C Signature Schema:2001, *XML Signature Schema*, W3C Recommendation, Copyright © [1 March 2001] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). <http://www.w3.org/TR/xmldsigcore/xmldsig-core-schema.xsd>

[b-XML Structures] W3C XML Schema Part 1:2001, *XML Schema Part 1: Structures*, W3C Recommendation, Copyright © [2 May 2001] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). <http://www.w3.org/TR/2001/RECxmlschema-1-20010502/>

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedia
Serie I	Red digital de servicios integrados
Serie J	Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedia
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	Gestión de las telecomunicaciones, incluida la RGT y el mantenimiento de redes
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Terminales y métodos de evaluación subjetivos y objetivos
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, 2009