UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T Q.775

(06/97)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE Q: CONMUTACIÓN Y SEÑALIZACIÓN

Especificaciones del sistema de señalización N.° 7 – Parte aplicación de capacidades de transacción

---

Directrices para la utilización de capacidades de transacción

Recomendación UIT-T Q.775

(Anteriormente Recomendación del CCITT)

RECOMENDACIONES DE LA SERIE Q DEL UIT-T

CONMUTACIÓN Y SEÑALIZACIÓN

SEÑALIZACIÓN EN EL SERVICIO MANUAL INTERNACIONAL	Q.1–Q.3
EXPLOTACIÓN INTERNACIONAL SEMIAUTOMÁTICA Y AUTOMÁTICA	Q.4–Q.59
FUNCIONES Y FLUJOS DE INFORMACIÓN PARA SERVICIOS DE LA RDSI	Q.60–Q.99
CLÁUSULAS APLICABLES A TODOS LOS SISTEMAS NORMALIZADOS DEL UIT-T	Q.100–Q.119
ESPECIFICACIONES DE LOS SISTEMAS DE SEÑALIZACIÓN N.° 4 Y N.° 5	Q.120–Q.249
ESPECIFICACIONES DEL SISTEMA DE SEÑALIZACIÓN N.° 6	Q.250–Q.309
ESPECIFICACIONES DEL SISTEMA DE SEÑALIZACIÓN R1	Q.310–Q.399
ESPECIFICACIONES DEL SISTEMA DE SEÑALIZACIÓN R2	Q.400–Q.499
CENTRALES DIGITALES	Q.500–Q.599
INTERFUNCIONAMIENTO DE LOS SISTEMAS DE SEÑALIZACIÓN	Q.600–Q.699
ESPECIFICACIONES DEL SISTEMA DE SEÑALIZACIÓN N.° 7	Q.700–Q.849
Generalidades	Q.700
Parte transferencia de mensajes	Q.701–Q.709
Parte control de la conexión de señalización	Q.711–Q.719
Parte usuario de telefonía	Q.720–Q.729
Servicios suplementarios de la RDSI	Q.730–Q.739
Parte usuario de datos	Q.740–Q.749
Gestión del sistema de señalización N.° 7	Q.750–Q.759
Parte usuario de la RDSI	Q.760–Q.769
Parte aplicación de capacidades de transacción	**Q.770–Q.779**
Especificaciones de las pruebas	Q.780–Q.799
Interfaz Q3	Q.800–Q.849
SISTEMA DE SEÑALIZACIÓN DIGITAL DE ABONADO N.° 1	Q.850–Q.999
RED MÓVIL TERRESTRE PÚBLICA	Q.1000–Q.1099
INTERFUNCIONAMIENTO CON SISTEMAS MÓVILES POR SATÉLITE	Q.1100–Q.1199
RED INTELIGENTE	Q.1200–Q.1999
RED DIGITAL DE SERVICIOS INTEGRADOS DE BANDA ANCHA (RDSI-BA)	Q.2000–Q.2999

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

RECOMENDACIÓN UIT-T Q.775

DIRECTRICES PARA LA UTILIZACIÓN DE CAPACIDADES DE TRANSACCIÓN

Resumen

La Recomendación Q.775 revisada proporciona directrices adicionales a usuarios TC al definir elementos de servicio de aplicación de usuario TC.

Orígenes

La Recomendación UIT-T Q.775, ha sido revisada por la Comisión de Estudio 11 (1997-2000) del UIT-T y fue aprobada por el procedimiento de la Resolución N.° 1 de la CMNT el 5 de junio de 1997.

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 Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), 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 N.° 1 de la CMNT.

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.

PROPIEDAD INTELECTUAL

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

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido/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.

ÍNDICE

Recomendación Q.775

DIRECTRICES PARA LA UTILIZACIÓN DE CAPACIDADES DE TRANSACCIÓN

(revisada en 1997)

1 Introducción

1.1 Generalidades

El objeto de esta Recomendación es proporcionar directrices a los posibles usuarios de capacidades de transacción (usuarios TC). Se facilitan a título ilustrativo solamente, ejemplos que indican la manera en que una aplicación puede utilizar la parte aplicación de capacidades de transacción (TCAP, *transaction capabilities application part*), pero no la manera en que las TC deben emplearse en todos los casos. La base técnica de esta Recomendación la constituyen las Recomendaciones Q.771 a Q.774, que deben considerarse como referencia primaria en caso de divergencia.

El objeto principal de las TC es proporcionar un soporte para aplicaciones interactivas en un entorno distribuido. Las TC se basan en el concepto de operaciones a distancia definido en la Recomendación X.880 (ROS, *remote operations*) junto con algunas mejoras y adiciones específicas del entorno del sistema de señalización N.° 7 para proporcionar los servicios que necesitan los usuarios TC. Las interacciones entre las entidades de aplicación distribuida se modelan mediante operaciones. Una operación se invoca mediante una entidad (de origen), en tanto que la otra entidad (de destino) intenta la ejecución de la operación y posiblemente devuelve el resultado de esta tentativa.

La semántica de una operación (representada por su nombre y sus parámetros) no reviste interés para las TC, las cuales proporcionan facilidades independientes de cualquier operación particular. Cuando defina una aplicación, el usuario TC deberá:

1) seleccionar operaciones (lo que supone definir la semántica y la sintaxis de los datos intercambiados durante las operaciones de invocación y sus respuestas);
2) seleccionar facilidades de las TC para soportar esas operaciones. Tales facilidades comprenden el tratamiento de operaciones individuales y la capacidad de disponer de cierto número de operaciones conexas anexadas a una asociación entre usuarios TC, denominada diálogo;
3) definir el "guión" (script) de la aplicación (por ejemplo, cuál de los dos pares invoca cada una de las operaciones, el orden de intercambio de mensajes que constituye el diálogo entre los pares de usuarios TC, y sus reacciones ante situaciones anómalas).

En esta Recomendación se describe el proceso de selección para la definición y utilización de las operaciones. Las operaciones que aparecen en lo que sigue son ficticias y se muestran con fines de ilustración solamente. Se describen, asimismo, las facilidades ofrecidas por las TC para el tratamiento de una operación o de una secuencia de operaciones en un diálogo. La definición de secuencias específicas de operaciones pertenece a la definición del protocolo de aplicación, que cae fuera del ámbito de esta Recomendación. No obstante, en las cláusulas 4 y 5 se proporciona una breve indicación de qué tipo de información debería contener una especificación de aplicación.

Los servicios TC son accesibles a los usuarios TC por conducto de primitivas. Estas primitivas modelan la interfaz entre TC y sus usuarios, pero no limitan la implementación de esta interfaz.

1.2 Entorno

Las TC define el protocolo de extremo a extremo entre usuarios TC ubicados en una red del sistema de señalización N.º 7. Actualmente no existe ninguna interfaz normalizada para el uso de las TC en ninguna otra red o protocolo subyacente (por ejemplo, X.25) distinto del SCCP del sistema de señalización N.º 7.

TC considera los usuarios que son sensibles al tiempo real y que no necesitan intercambiar grandes cantidades de datos. Se considera que para este tipo de usuarios, los protocolos normalizados definidos por las capas 4 a 6 de la OSI en las Recomendaciones de la serie X producirán taras (overheads) excesivas, por lo que no se utilizan.

En consecuencia, las TC no pueden soportar todas las clases de aplicaciones y algunas de éstas necesitarán apoyarse en servicios más elaborados, tales como los especificados en las Recomendaciones de la serie X. Además de expresar lo que puede efectuar las TC, esta Recomendación indica lo que es incapaz de realizar, a fin de ayudar al diseñador de aplicaciones a elegir la manera de soportar una aplicación.

2 Operaciones

2.1 Definición

Un usuario TC de origen invoca una operación para pedir a un usuario TC de destino que realice una acción determinada.

Cada operación pertenece a una clase particular, que indica si el destino debe o no comunicar resultado con éxito (result), sin éxito (error), o ambos, o ninguno.

La clase de una operación no se señala al usuario TC distante cuando se invoca la operación; se supone que en ambos extremos la aplicación posee una comprensión común de la clase de cada operación en curso.

Además de la clase, la definición de una operación incluye un valor de temporización que indica el tiempo máximo en el que debe completarse la operación y el resultado comunicado.

Este valor de temporizador es un asunto local. No se comunica al extremo distante a través de ningún protocolo. Lo elige el usuario TC cuando define la operación en base a las expectativas del tiempo de propagación de ida y retorno de un usuario TC a otro y de las demoras del procesamiento.

Una operación se define por:

– su código de operación y el tipo de los parámetros asociados con la petición de operación;
– su clase;
– si la clase requiere un informe de éxito, los posibles resultados correspondientes a las ejecuciones exitosas se definen mediante una lista de parámetros;
– si la clase requiere un informe de fallo, los posibles resultados correspondientes a situaciones en que el usuario TC distante no pudo ejecutar completamente la operación. Cada una de estas situaciones se identifica mediante una causa de error específica; la lista de estas causas de error forma parte de la definición de la operación. Puede añadirse información de diagnóstico a la causa de error: si está presente, forma parte de la definición;
– la lista de las posibles operaciones enlazadas (linked operations), si se admiten respuestas consistentes en operaciones enlazadas para esa operación. Las operaciones enlazadas deberán describirse separadamente;

– un valor de temporizador que indique el intervalo en el cual debe completarse la operación y devolverse, caso de existir. Este valor de temporización puede ser uno de los factores utilizados por una implementación para gestionar el ID de invocación asociado a la invocación de la operación. (Cuando expira el temporizador asociado a una invocación de operación, se devuelve el ID de invocación al conjunto de ID de invocación tras un periodo de "congelación" adecuado que depende de la implementación.)

Por regla general, la elección de la clase de operación debe basarse en la semántica asociada a la operación, y las operaciones no deben tener que cursarse en un tipo de mensaje en particular. Por ejemplo, si el invocador de la operación no requiere un acuse de que ésta ha podido llevarse a cabo o no, una operación de clase 4 puede ser la más apropiada. Si el invocador de una operación no necesita un conocimiento explícito de que la misma se ha realizado con éxito pero desea saber si no se la ha podido realizar en absoluto, es conveniente utilizar una operación de clase 2. Por ejemplo, la definición de la operación "Anuncio" como de clase 2 ó 4 sugiere distintas intenciones del invocador de la operación, aunque las acciones del realizador de la operación sean idénticas.

2.2 Ejemplos

2.2.1 Tratamiento de operaciones simples

NOTA – La invocación de la operación, así como el informe de resultado con éxito debe caber en un mensaje. Los informes de éxito pueden segmentarse utilizando retorno de resultado-no último y retorno de resultado-último.

Clase 1 (se informan tanto el éxito como el fallo)

Traducir un número con franquicia a un número de abonado llamado; devolver el número llamado si puede efectuarse la traducción. En otro caso indicar por qué no es así; tiempo atribuido: 2 segundos.

Si no se devuelve una respuesta a una invocación de operación una vez expirado el temporizador, se informa al usuario TC (cancelación de la operación por la TC); éste puede suponer que la invocación o la respuesta se ha perdido y, según los requisitos de la operación, tomar las medidas correctivas adecuadas (por ejemplo, invocar nuevamente la operación, informar a la gestión local, etc.).

Clase 2 (sólo se informa el fallo)

Realizar una prueba de rutina y enviar una respuesta solamente en el caso en que exista algún fallo; tiempo atribuido: 1 minuto.

En el caso de una operación de clase 2, si no se recibe ningún resultado cuando expira el temporizador, se informa al usuario TC. Esto se interpreta como un resultado con éxito, aun cuando se pierda la invocación.

Cuando se seleccione la clase 2 deberá contemplarse este aspecto.

Clase 3 (sólo se informa el éxito)

Realizar una prueba: esto corresponde a un punto de vista pesimista en que se considera el fallo como opción por defecto que no requiere ninguna respuesta.

Se informa al usuario TC sobre la expiración del temporizador: el usuario TC deberá interpretar esta información como un fallo de la operación (aunque esto es considerado normal por el TC, que estima que la operación ha terminado). Al seleccionar la clase 3 deberá considerarse este aspecto.

Clase 4 (no se informa ni el éxito ni el fallo)

Enviar un aviso, sin esperar una respuesta ni acuse de recibo de ningún tipo.

En este caso, no se desprende ningún resultado de la invocación de la operación. El usuario TC confía en la TC y en la red para la entrega de la invocación. La notificación de la expiración del temporizador es un asunto local.

Comparación con las clases de operación ROSE (Recomendación X. 219)

El ROSE establece cinco clases de operaciones: las clases 2 a 5, denominadas clases asíncronas, son idénticas a las clases 1 a 4 de la TC. La clase 1 del ROSE es una clase síncrona que no tiene correspondencia en la TC, donde se consideran intercambios dúplex de componentes.

Sin embargo, un usuario TC puede decidir operar en un modo síncrono (véase 3.2.1).

2.2.2 Tratamiento de operaciones más complejas

Operaciones con resultados segmentados

Un resultado exitoso puede dividirse en varios segmentos, cada uno de los cuales se indica al originador de la aplicación mediante una primitiva. Los usuarios TC pueden emplear esta facilidad, que utiliza la primitiva TC-RESULTADO-NL, para remediar la ausencia de segmentación en las capas subyacentes. El último segmento se indica mediante la primitiva TC-RESULTADO-L.

El informe de un error no puede ser segmentado.

Cuando el diseñador del protocolo puede asegurar que la segmentación la proporcionan las capas subyacentes a través de las rutas de señalización por las que se transfieren los mensajes TC, se desaconseja el empleo de esta facilidad de segmentación.

En el caso de un resultado segmentado, la TC no puede identificar un segmento específico.

El usuario TC debe asegurar que se pueda pasar cada segmento (es decir, el parámetro Parámetro de cada primitiva petición TC-RESULTADO-NL/L debe contener información suficiente como para permitir la construcción de un valor válido del tipo (o subtipo compatible) asociado con el resultado de la operación.

Ejemplo E1: Una operación solicita la ejecución de una prueba. El resultado de una ejecución correcta se segmenta en tres partes P1, P2 y P3, que se devuelven al originador.

Una posible secuencia de primitivas para el ejemplo E1 se indica en el cuadro 1.

Cuadro 1/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (Prueba) Petición TC-RESULTADO-NL (P1)
Indicación TC-RESULTADO-NL (P1)	Petición TC-RESULTADO-NL (P2)
Indicación TC-RESULTADO-NL (P2)	Petición TC-RESULTADO-L (P3)
Indicación TC-RESULTADO-L (P3)	
Tiempo	
NOTA – El valor de la temporización es especificado por el usuario TC de origen en el momento de la invocación. Un resultado no definitivo no hace rearrancar la temporización.	

Operaciones enlazadas

Otro elemento del método de operación básico es la capacidad de enlazar una invocación de operación con otra invocación de operación.

Normalmente esta facilidad abarca situaciones en que el destino de la operación original (con la que se efectúa el enlace) requiere información adicional para procesar esta operación: tal es el caso cuando se utilizan facilidades de menú (las facilidades de menú permiten a un usuario efectuar una secuencia de elecciones, cada una de las cuales depende de las anteriores).

Ejemplo E2: La operación consiste en la ejecución de una prueba con varias opciones. Antes de ejecutarse la prueba, se ofrecen esas opciones al originador de la misma para que efectúe una selección entre ellas (usuario TC A). Se entrelazan dos operaciones: la operación 1 es la prueba y la operación 2 es la de selección de la opción: el usuario TC A responde en primer lugar a la operación 2 antes de que el usuario TC B pueda efectuar la prueba con la opción u opciones indicadas.

Una posible secuencia de primitiva para el ejemplo E2 se indica en el cuadro 2.

No hay ningún límite al número de invocaciones de operación que pueden enlazarse con una invocación de operación dada.

Debe observarse que, cuando una operación B está enlazada con otra operación A, éstas no necesitan estar anidadas (nested). La única condición es que la invocación de B tenga lugar antes de que se notifique la consecuencia de A. Sin embargo, la operación B no tiene por qué terminar con anterioridad a la operación A.

Cuadro 2/Q.775

Usuario TC A	Usuario TC B	
Petición TC-INVOCACIÓN (Prueba, Clase = 1)		Comienzo de la operación 1
	Indicación TC-INVOCACIÓN (Prueba)	
	Petición TC-INVOCACIÓN (Selección de opción, Clase = 1)	Comienzo de la operación 2
Indicación TC-INVOCACIÓN (Selección de opción)		
Petición TC-RESULTADO-L (Opciones)	Indicación TC-RESULTADO-L (Opciones)	Fin de la operación 2
	Petición TC-RESULTADO-L (Resultado de la prueba)	
Indicación TC-RESULTADO-L (Resultado de la prueba)		Fin de la operación 1
Tiempo		

2.3 Facilidades relacionadas con los componentes ofrecidas a los usuarios TC

2.3.1 Invocación

Hasta ahora, se han considerado las operaciones desde el punto de vista estático. La invocación introduce un aspecto dinámico: debe diferenciarse la invocación específica de una operación de otras posibles invocaciones concurrentes de la misma operación o de otra cualquiera.

Cada activación particular de una operación se identifica mediante un ID de invocación. Este ID de invocación debe ser inequívoco. Es seleccionado por el usuario TC que origina la invocación de la

operación y es transferido al usuario TC de destino que lo reflejará en su respuesta (o en cada segmento de respuesta) o en una invocación conexa: en consecuencia, correlaciona la respuesta a una invocación (o cada segmento de una respuesta) o una invocación conexa con la propia invocación.

El usuario TC tiene libertad para asignar cualquier valor al ID de invocación (índice, dirección, ...), siempre que su valor pueda hacerse corresponder a un entero que pueda codificarse en un octeto de acuerdo con las reglas de codificación especificadas en la Recomendación Q.773. Obsérvese que dicho entero toma valores entre –128 y 127.

El ID de invocación asociado con una invocación puede reutilizarse cuando se ha recibido el último o único segmento de un resultado, o cuando la TC indique situaciones anormales. Sin embargo, no debe reatribuirse el valor inmediatamente para otra activación de operación ya que la reatribución inmediata impediría el tratamiento correcto de algunas situaciones (véase más adelante).

Se denomina periodo de congelación al periodo durante el cual un ID de invocación está liberado pero no puede ser reatribuido. Este periodo depende de la implementación.

Como los ID de invocación reciben su valor dinámicamente en el momento en que se invoca la operación, dicho valor no puede aparecer en la especificación de los protocolos de aplicación. Se trata más bien de indicar un valor "lógico", que debe sustituirse por un valor real en el momento de la ejecución, a fin de identificar una operación en un flujo único.

Teniendo en cuenta los ID de invocación, la secuencia de primitivas para el ejemplo E2 anterior sería tal como se indica en el cuadro 3.

Cuadro 3/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-INVOCACIÓN (2, 1, Selección de opción, Clase = 1)
Indicación TC-INVOCACIÓN (2, 1, Selección de opción) Petición TC-RESULTADO-L (2, Opciones)	Indicación TC-RESULTADO-L (2, Opciones) Petición TC-RESULTADO-L (1, Resultado de la prueba)
Indicación TC-RESULTADO-L (1, Resultado de la prueba)	
Tiempo	

donde el primer parámetro de una primitiva indica un ID de invocación. Cuando deban estar presentes ambos parámetros, el segundo de ellos es el ID enlazado. Esto no es más que un convenio de notación.

2.3.2 Cancelación (por el usuario TC)

El usuario TC que pide la invocación de una operación puede parar la actividad asociada al ID de invocación correspondiente, por cualquier motivo que considere apropiado. Sin embargo, en principio, deberá reservarse la cancelación para situaciones anormales. El método normal de terminación de una operación es la recepción de un resultado o la finalización basada en la expiración de un temporizador.

La cancelación tiene solamente efecto local: no impide que el usuario TC distante envíe respuestas a una operación cancelada. Cuando se reciban tales respuestas, serán rechazadas por la TC como se indica en el ejemplo que sigue, que representa una secuencia de primitivas para el ejemplo E1 definidas anteriormente, en el caso en que el usuario TC A cancele la prueba tras la recepción del primer segmento del resultado.

En el cuadro 4, el segmento P2 no lo recibe el usuario TC A: la TC detecta una situación de rechazo (ID de invocación no activo) y por consiguiente no lo entrega al usuario TC A, por lo que en el lado A se rechaza todo intento del usuario TC B de enviar más segmentos de respuesta.

Cuadro 4/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-RESULTADO-NL (1, P1)
Indicación TC-RESULTADO-NL (1, P1) Decisión de cancelación: Petición TC-CANCELACIÓN (1) Indicación TC-RECHAZO (1, Código de problema) ...	Petición TC-RESULTADO-NL (1, P2) ...
Tiempo	

2.3.3 Rechazo (por el usuario TC)

El usuario TC es el único responsable de decidir cuándo debe enviarse un componente de rechazo o devolverse una indicación de error (fallo en la realización de una operación). El usuario TC puede rechazar un componente por cualquier razón que considere adecuada, siempre que en las especificaciones de la TC haya un código de problema de rechazo adecuado (por ejemplo, falta del elemento de información obligatorio en una operación, error o respuesta, operación inesperada, operación desconocida, etc.) que pueda utilizar al efecto.

De forma similar, el usuario TC decide qué código de error e información de diagnóstico (especificados como parte de las especificaciones del protocolo de usuario TC y acordados por los dos usuarios TC pares con este fin preciso) ha de utilizarse cuando se envíe un componente de error.

El cometido del usuario TC se ilustra mediante el ejemplo siguiente.

El usuario TC del lado A espera, en una situación determinada, recibir la operación Y únicamente como una operación enlazada. Al recibir del lado B un componente de invocación con un código de operación referido a Y pero ningún ID enlazado, el usuario TC del lado A puede elegir entre:

– no realizar la operación y devolver un error con algún parámetro de diagnóstico previamente determinado, especificado en la especificación de aplicación de usuario TC a tal efecto;
– rechazar el componente como una "operación no reconocida".

La interpretación del diagnóstico de error devuelto o del código de problema de rechazo corresponde al usuario TC del lado B y no se describe en las Recomendaciones relativas a la TC.

El rechazo de una invocación de operación o de un resultado afecta a la totalidad de la operación: para esta invocación el TC no aceptará más respuestas. El rechazo de una operación enlazada no

afecta a la operación conexa por lo que respecta al TC. Los usuarios TC deben describir sus reacciones a dichas situaciones anormales como parte de su guión de aplicación.

Esto se ilustra en el cuadro 5 en la que, en el ejemplo E2, el usuario TC A no espera el proceso de selección de opción (puede tratarse de una característica facultativa) y rechaza la operación con el código de problema "operación enlazada inesperada". El usuario TC B, puede decidir entonces la ejecución de la prueba adoptando una opción por defecto.

Cuadro 5/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-INVOCACIÓN (2, 1, Selección de opción, Clase = 1)
Indicación TC-INVOCACIÓN (2, 1, Selección de opción) Petición TC-RECHAZO-U (2, Código de problema)	
	Indicación TC-RECHAZO-U (2, Código de problema) Petición TC-RESULTADO-L (1, Resultado de la prueba)
Indicación TC-RESULTADO-L (1, Resultado de la prueba)	
Tiempo	

Cuando se rechaza una invocación de operación, el usuario TC puede decidir una nueva invocación (por ejemplo, en caso de corrupción del componente de invocación); esto podría ser una nueva invocación (nuevo ID de invocación). El usuario puede tomar también la decisión de abortar el diálogo. Un diálogo muy simple (una pregunta y una respuesta) puede no definir ningún tipo de mecanismo de recuperación, salvo cuando se trate de una operación de importancia esencial (por ejemplo, actualización de una base de datos).

2.3.4 Cancelación distante (por el usuario TC)

TC no proporciona ningún servicio específico para cancelar la ejecución a distancia de una operación en curso. El servicio de cancelación proporcionado por la primitiva petición TC-U-CANCELACIÓN tiene únicamente un efecto local (véase 2.3.2).

Sin embargo, puede definirse un procedimiento de cancelación a distancia a nivel de usuario TC, utilizando los servicios TC existentes. Una solución para el usuario TC es incluir en uno de los ASE utilizados por el contexto de aplicación una operación cuyo objetivo sea cancelar las invocaciones existentes.

El siguiente módulo ASN.1 proporciona una descripción de dicho tipo de operación. Este tipo (y el tipo de error asociado) puede importarse en uno de los módulos utilizados por el usuario TC, de tal forma que éste pueda adjudicar los códigos de operación y error adecuados. De forma alternativa, el usuario TC puede diseñar también su propia operación de acuerdo con los mismos principios.

```
TCAP-Tools { ccitt recommendation q 775 modules(2) tools(1) version1(1) }

DEFINITIONS ::=
BEGIN
EXPORTS Cancel, CancelFailed, Cancelled;

IMPORTS OPERATION, ERROR, InvokeIdType
FROM TCAPMessages { ccitt recommendation q 773 modules(2) messages(1) version2(2) };

Cancel ::= OPERATION
ARGUMENT InvokeIdType
-- a TC-user may redefine this type to include an empty result so that it becomes a Class 1 operation
ERRORS { CancelFailed }
-- timer = 15 s

CancelFailed ::= ERROR
PARAMETER SET {
        problem  [0] CancelProblem,
        invokeId  [1] InvokeIdType }

CancelProblem ::= ENUMERATED
{ unknownInvocation(0),
        tooLate (1),
        notCancellable (2) }
-- a TC-user may redefine this type to include application-specific problems

Cancelled ::= ERROR
-- an error of this type should be included in the error list of cancellable operations

END
```

Es necesario incluir un error "cancelado" en la lista de errores adjunta a las operaciones cancelables. En tal caso, el usuario TC que recibe la petición de cancelación emitirá una primitiva petición TC-U-ERROR para finalizar la operación. La recepción de un componente de error con este código de error y el correspondiente indicador TC-U-ERROR finalizará la operación en el lado de invocación.

La operación que va a cancelarse se identifica mediante el indicador invocación (invokeId) asignado al mismo en el momento de la invocación. La invocación de la operación cancelación no afecta a la máquina de estados de invocación de la operación que va a cancelarse porque el indicador invocación cursado en el argumento de la operación no es visible para la subcapa del componente.

Si falla la cancelación, se informa de un error de usuario con tres posibles diagnósticos:

– unknownInvocation (invocación desconocida): Si la invocación nunca ha tenido lugar o ha sido olvidada;

– tooLate (demasiado tarde): Si la invocación aún se conoce pero la ejecución se encuentra en una etapa que no permite la cancelación;

– notCancellable (no cancelable): El indicador invocación en el argumento de la operación de cancelación corresponde a una operación que no ha sido acordada por los usuarios TC como cancelable por el lado de invocación.

Cuando la cancelación se lleva a cabo con éxito y no se recibe de la operación cancelada ningún error de retorno con código de error que indique "cancelado", expirará el temporizador asociado con la operación cancelada, lo que hará que el TC emita una primitiva indicación TC-L-CANCELACIÓN.

De forma alternativa, el usuario TC que solicita la cancelación puede decidir emitir una petición TC-U-CANCELACIÓN inmediatamente después de enviar la petición de cancelación, de tal forma que ya no exista ninguna actividad local para la operación que va a cancelarse.

La utilización del mecanismo de cancelación tiene sentido si la operación cancelación se invoca antes de que expire el temporizador asociado con la operación que va a cancelarse. Tras ello el indicador invocación puede no ser reconocido en el lado de realización, puesto que puede haberse completado la ejecución de la operación y emitido la primitiva de respuesta. La posibilidad de cancelar la ejecución de una operación tras la expiración del temporizador escapa al ámbito de estas directrices, puesto que significaría que el objetivo no es cancelar la ejecución de la operación sino las acciones subsiguientes que pueden haber sido desencadenadas en el extremo de realización por la invocación.

2.3.5 Reinicio del temporizador de la operación por el usuario TC

La selección de un valor adecuado del temporizador forma parte de la definición de una operación. La TC inicia la supervisión del temporizador cuando se envía el componente de invocación con el que está relacionado. Sin embargo, el usuario TC tiene la posibilidad de pedir a la TC que reinicie dicho temporizador en cualquier momento antes de que expire.

El diseñador del protocolo selecciona el valor del temporizador de acuerdo con una estimación del tiempo requerido para completar la ejecución de la operación y transferir el resultado. Sin embargo, hay casos en los que puede preverse una dispersión significativa de los tiempos reales de ejecución, haciendo difícil la selección de un valor adecuado.

Para evitar que el temporizador de la operación expire mientras la petición está aún realizándose, el diseñador del protocolo puede elegir uno de los dos enfoques siguientes:

a) seleccionar un valor del temporizador que sea mayor que el tiempo estimado para que se complete la operación en el peor de los casos;

b) seleccionar un valor del temporizador que corresponda con el caso más probable, y disponer el usuario TC de tal forma que pueda utilizar el servicio TC-REINICIO-TEMPORIZADOR cuando detecte que la operación tardará más de lo esperado en completarse. En este caso, deberán definirse los procedimientos adecuados para asegurar que el usuario TC es informado de esta situación (por ejemplo, utilizando operaciones enlazadas).

En el ejemplo siguiente, el usuario TC A invoca varias veces la operación "interrogación" para recuperar tres tipos distintos de información de un sistema de bases de datos que está distribuido en dos ubicaciones B1 y B2. Dado que el usuario TC A no conoce cómo están distribuidos los datos en dichas dos ubicaciones, envía todas las peticiones al usuario TC en B1.

En el primer caso, la información solicitada está disponible localmente en B1. En los otros dos casos, la información solicitada debe de recuperarse de B2. El usuario TC B1 notifica al usuario TC A que completar la petición llevará más tiempo del esperado, invocando una operación "espera". En el segundo caso, el usuario TC A acepta esperar más de lo previsto y solicita a la TC que reinicie el temporizador de la operación. En el tercer caso, el usuario TC A prefiere abandonar la operación con un procedimiento específico del usuario similar a los descritos en 2.3.4. Véase el cuadro 5 *bis*.

Cuadro 5 *bis*/Q.775

Usuario TC A	Usuario TC B1	Usuario TC B2
Primera interrogación Petición TC-INVOCACIÓN (1, interrogación, Expiración = 5 s)	Indicación TC-INVOCACIÓN (1, interrogación) Petición de información localmente disponible Petición TC-RESULTADO-L (1, interrogación-resultado)	
Indicación TC-RESULTADO-L (1, interrogación-resultado) Segunda interrogación Petición TC-INVOCACIÓN (2, interrogación, Expiración = 5 s)	Indicación TC-INVOCACIÓN (2, interrogación) Petición de información no disponible localmente Petición TC-INVOCACIÓN (3, 2, espera) Petición TC-INVOCACIÓN (1, interrogación)	
Indicación TC-INVOCACIÓN (3, 2, espera) Petición TC-REINICIO-TEMPORIZADOR (2)		Indicación TC-INVOCACIÓN (1, interrogación) Petición TC-RESULTADO-L (1, interrogación-resultado)
	Indicación TC-RESULTADO-L (1, interrogación-resultado) Enviar resultado recibido a A Petición TC-RESULTADO-L (2, interrogación-resultado)	
Indicación TC-RESULTADO-L (2, interrogación-resultado) Tercera interrogación Petición TC-INVOCACIÓN (3, interrogación, Expiración = 5 s)	Indicación TC-INVOCACIÓN (3, interrogación) Petición de información no disponible localmente Petición TC-INVOCACIÓN (4, 3, espera) Petición TC-INVOCACIÓN (2, interrogación)	

Cuadro 5 *bis*/Q.775 (*fin*)

Usuario TC A	Usuario TC B1	Usuario TC B2
Indicación TC-INVOCACIÓN (4, 3, espera) Petición TC-INVOCACIÓN [5, cancelación (3)]		Indicación TC-INVOCACIÓN (2, interrogación) Petición TC-RESULTADO-L (2, interrogación-resultado)
	Indicación TC-INVOCACIÓN [5, cancelación (3)] Petición TC-U-ERROR (3, cancelado) Indicación TC-RESULTADO-L (1, interrogación-resultado) Descartar resultado recibido en A	
Indicación TC-U-ERROR (3, cancelado)		

2.4 Situaciones anormales relacionadas con los componentes

2.4.1 Pérdida de un componente

La TC presupone una probabilidad de pérdida de mensajes en la red muy pequeña. Si esta probabilidad es excesiva para una aplicación, deberá utilizarse el método de servicio de red con conexión. Si algunas informaciones de protocolo requieren una calidad de servicio mejorada (por ejemplo, información de tasación) la aplicación deberá establecer sus propios mecanismos para lograr una mayor fiabilidad para esta información.

Pérdida de una invocación de operación

En el cuadro 6 se ilustra el caso correspondiente al ejemplo E1, en que no se recibe una respuesta a la prueba antes de la expiración del temporizador.

Cuadro 6/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	
Límite de tiempo: Indicación TC-CANCELACIÓN-L (1)	
Tiempo	

Cuando se pierde una operación de clase 1, se informa al usuario TC de cuándo expira el temporizador asociado con la operación. Cuando se pierde una operación de clase 1 con un solo resultado, la TC no puede indicar si se ha perdido la invocación de operación o la respuesta. Si la aplicación debe establecer una discriminación entre estos dos casos, deberá hacerlo en el protocolo de aplicación (por ejemplo empleando una marca temporal o mediante acuse de recibo de la invocación de operación, antes de responder a la misma).

Para una operación de clase 2, la pérdida se considerará como un éxito (tanto si se perdió la invocación como el informe de fallo). Teniendo en cuenta la probabilidad de pérdida, esto puede ser aceptable para operaciones que no sean fundamentales (por ejemplo, mediciones estadísticas).

En el caso de una operación de clase 3, el tratamiento que se da a la pérdida es el mismo que el de un fallo de operación, tanto si se ha perdido la invocación como el informe de éxito.

Para una operación de clase 4, la pérdida no será visible por la TC.

Pérdida de un resultado

– La TC no detecta nunca la pérdida de un resultado no final.

– La pérdida de un resultado final se indicará finalmente al usuario TC cuando se alcance el límite de tiempo, pero no puede siempre interpretarse de forma inequívoca como pérdida de respuesta. Si no se ha recibido ningún resultado no final, ello puede deberse a que se haya perdido la invocación.

Pérdida de una operación enlazada

La pérdida de una operación enlazada surte el mismo efecto que la pérdida de una operación no enlazada. No tiene ningún efecto sobre la operación con la que se realiza el enlace.

Pérdida de un componente de rechazo

Este caso tendrá una frecuencia de aparición muy reducida por lo que ninguna aplicación intentará recuperarse de tal situación. Si la pérdida del rechazo afecta a una invocación de operación, cuando expire la temporización de la operación, el usuario TC que invocó la operación estimará que dicha invocación (o la respuesta) se ha perdido y reaccionará en consecuencia. Si se trata de una respuesta, el originador de la respuesta no sabrá que era incorrecta y la detección de la pérdida corresponderá al originador de la operación.

2.4.2 Duplicación de un componente

Como las duplicaciones de mensajes son muy raras en la red del sistema de señalización N.° 7, no es necesario que los guiones para aplicaciones del SS N.° 7 definan casos complejos en previsión de estas situaciones. Sin embargo, toda aplicación en que la duplicación fuera inaceptable, debería definir su propio mecanismo de detección de duplicación o utilizar un servicio con conexión.

Invocación de operación duplicada

Cuando una invocación de operación es duplicada (por el proveedor del servicio), el usuario TC de destino (B) puede o no detectar la operación:

– El usuario TC B detecta la duplicación: el duplicado puede rechazarse utilizando el código de problema "ID de invocación de duplicado". En este caso, el rechazo puede ser interpretado por el usuario TC distante como un rechazo de la invocación original.

– El usuario TC B no detecta la duplicación: esto puede ocurrir cuando existe una relación de amo-esclavo entre A y B, y B ejecuta la operación con desconocimiento del contexto.

Suponiendo, en el ejemplo E1, que se trata del segundo caso, una secuencia posible podría ser la indicada en el cuadro 7.

Cuadro 7/Q.775

Usuario TC A	Usuario TC B	
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba) Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-RESULTADO-NL (1, P1) Petición TC-RESULTADO-NL (1, P1)	Duplicación de la invocación no detectada
Indicación TC-RESULTADO-NL (1, P1) Indicación TC-RESULTADO-NL (1, P1) A detecta una situación anormal y la rechaza: Petición TC-RECHAZO-U (1, Código de problema) TC detecta una situación anormal y rechaza P2: Indicación TC-RECHAZO-L (1, Código de problema)	Petición TC-RESULTADO-NL (1, P2) Indicación TC-RECHAZO-U (1, Código de problema)	
	Indicación TC-RECHAZO-R (1, Código de problema)	
Tiempo		

En esta secuencia, el usuario TC B considera dos invocaciones de prueba independientes y responde a cada una de ellas. Se acepta el primer resultado P1. El usuario TC A detecta que P1 se ha recibido por segunda vez y lo rechaza. Esto concluye la operación y origina el rechazo del resultado P2 cuando se reciba (rechazado por la TC). En consecuencia, en el lado B ambas actividades concluirán con la recepción de los rechazos.

Duplicación de un resultado no final

La TC no puede detectar la duplicación de un resultado no final, por lo que lo entregará dos veces al usuario TC. La detección de esta situación se deja para la aplicación.

Duplicación de un resultado final

En caso de duplicación de un resultado final (RR-L), la TC puede detectar la situación. El segundo resultado final será considerado como anormal (la operación ha finalizado con el primer resultado "final") y la TC la rechaza.

En el cuadro 8 se indica una secuencia correspondiente al ejemplo E1 en el que el tercer segmento del resultado es duplicado (por la red).

Comentario: Podría parecer una característica interesante que la TC descartase los duplicados en todos los casos. Sin embargo, debe observarse que:

1) esto exigiría otro grado de complejidad en la TC, que contradice las características básicas de la TC en el método sin conexión;

2) esto corresponde a una situación que es sumamente rara, al menos en la red del sistema de señalización N.º 7.

Para afrontar estas situaciones cuando lo requiera una aplicación, es mejor utilizar el método de servicio de red con conexión, ya que la duplicación puede ser detectada y tratada en las capas inferiores.

Cuadro 8/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1) Indicación TC-RESULTADO-NL (1, P1) Indicación TC-RESULTADO-NL (1, P2) Indicación TC-RESULTADO-L (1, P3) TC recibe RR-L (1, P3) Duplicación de P3: Indicación TC-RECHAZO-L (1, Código de problema)	Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-RESULTADO-NL (1, P1) Petición TC-RESULTADO-NL (1, P2) Petición TC-RESULTADO-L (1, P3)
	Indicación TC-RECHAZO-R (1, Código de problema)
Tiempo	

2.4.3 Secuenciación incorrecta de los componentes

El orden de los resultados segmentados no es importante para la TC. Si dicho orden es importante para el usuario TC, deberán definirse mecanismos apropiados en el protocolo de aplicación (por ejemplo, introduciendo un método de numeración para identificar las respuestas intermedias en un parámetro de estas respuestas, o empleando un servicio con conexión).

Debido a una secuenciación errónea, un resultado no final puede llegar después de un resultado final; en tal situación, la TC rechaza el resultado no final. Esto ocurre porque la recepción del resultado final obliga a la TC a cerrar la máquina de estados de invocación asociada con esta operación, y entonces, cuando se recibe el resultado no final demorado, no se le puede asociar con ninguna máquina activa de estados de invocación.

La secuencia ilustrada en el cuadro 9 se indica lo que ocurre en el ejemplo E1 cuando se recibe la última parte del resultado antes que la segunda: se informa a ambos usuarios TC.

Si se recibe una invocación de operación enlazada tras el resultado final de la operación con la que se efectúa el enlace (como consecuencia de una secuenciación errónea), se rechaza la operación enlazada.

La TC supone una probabilidad de secuenciación errónea muy reducida. Si la red de soporte no es satisfactoria a este respecto, deberá considerarse la utilización del servicio de red con conexión.

2.4.4 Rechazo de un componente por la TC

Cuando la TC recibe un componente (invocación de operación o respuesta) que no se ha formateado correctamente o se ha recibido fuera de contexto (por ejemplo, respuesta sin invocación de operación previa), un principio general es proceder a su rechazo, lo que significa que:

1) TC forma un componente rechazo para informar al originador del componente que falla e informa al usuario TC local del componente rechazo en espera del envío al extremo distante. La TC proporciona toda la información disponible sobre la naturaleza del componente rechazado (si el diálogo no ha sido ya terminado por el usuario TC distante);

2) como reacción, el usuario TC local puede tomar la decisión de abortar, continuar o finalizar el diálogo. En los dos últimos casos, cuando el usuario TC notifique su decisión a la TC, se informará del rechazo al usuario TC par.

Cuadro 9/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba) Petición TC-RESULTADO-NL (1, P1)
Indicación TC-RESULTADO-NL (1, P1) Indicación TC-RESULTADO-L (1, P3) Llega la PDU RR-NL (1, P2) Resultado con secuencia errónea: rechazo (ninguna máquina de estados activa) Indicación TC-RECHAZO-L (1, Código de problema)	Petición TC-RESULTADO-NL (1, P2) Petición TC-RESULTADO-L (1, P3)
	Indicación TC-RECHAZO-R (1, Código de problema)
Tiempo	

En las subcláusulas anteriores han aparecido los posibles casos de rechazo por la TC. Siempre que se reconozca el ID de invocación, el rechazo por la TC produce la terminación de la operación. Una posible recuperación por el usuario TC consistirá en una nueva invocación de la operación terminada. Cuando la componente rechazada no es identificable, se informa al usuario TC local del componente defectuoso recibido. Se incluye un valor ID de invocación NULO en el componente de rechazo en espera de envío. Sin embargo, como este componente de rechazo no puede asociarse con ninguna invocación conocida, la reacción apropiada puede ser la de abortar el diálogo.

2.4.5 Expiración del temporizador de operación

Cuando la TC informa al usuario TC sobre la expiración del temporizador (indicación TC-L-CANCELACIÓN), ello indica que no puede recibirse más información sobre la invocación de la operación (en especial, ningún rechazo). Si la entidad par envía todavía información en relación a esta invocación, se descartará dicha información cuando se reciba, siempre que no se haya reatribuido el ID de invocación de la operación cancelada. Se evita, generalmente la reatribución prematura de los valores de ID de invocación, ajustando correctamente los valores del temporizador y eligiendo el instante de "congelación" de un ID de invocación una vez liberado. A fin de compensar las incertidumbres del tiempo total necesario para enviar información desde un usuario TC a otro, sin tener en cuenta el caso más desfavorable absoluto (que es además, en general, el más improbable) se necesita el establecimiento de un mecanismo dependiente de la implementación, que evite la reatribución prematura de los ID de invocación.

La indicación de expiración de un temporizador corresponde a una situación anormal en el caso de las operaciones de clase 1 solamente. En ese caso, el usuario TC es consciente de que se ha perdido la invocación o la respuesta. Si no surgen efectos secundarios indeseables, puede producirse otra invocación de la misma operación tras la expiración del temporizador. Esto se indica, en el ejemplo E1, mediante la secuencia ilustrada en el cuadro 10.

Para una operación de clase 2, la expiración del temporizador indica que para esta invocación, no se ha recibido o no se aceptará ningún fallo. Se trata de una indicación definida de éxito (para la clase 2) si se supone que no hay posibilidad de pérdida de mensaje en la red. En caso de fallo, se aplica a la clase 3 una situación similar. La indicación de expiración del temporizador para una operación de clase 4 es una decisión local.

Cuadro 10/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Prueba, Clase = 1)	Indicación TC-INVOCACIÓN (1, Prueba)
Expiración del temporizador: Indicación TC-CANCELACIÓN-L (1) Petición TC-INVOCACIÓN (2, Prueba, Clase = 1)	
Tiempo	

3 Diálogos

Siempre que se emita una de las primitivas de tratamiento de operación contempladas en la cláusula 2, se cursa una petición a la TC, pero no se envía nada al usuario TC distante, hasta que se emita una primitiva de petición de transmisión. A continuación se tratará de estas primitivas y de su relación con las primitivas de tratamiento de operación.

3.1 Agrupación de componentes en un mensaje

El efecto de la emisión por parte de un usuario TC de una primitiva de tratamiento de componente (a menos que dicha primitiva tenga solamente efecto local) es la creación de un componente para incluirlo en un mensaje. El mensaje no se transmite hasta que el usuario TC lo solicite.

Obsérvese que puede también generarse un componente como consecuencia de un procedimiento de rechazo de la TC: en este caso, dicho componente se coloca en el siguiente mensaje de diálogo, salvo si éste es abortado.

Pueden agruparse varios componentes y enviarse al extremo distante como un único mensaje para ahorrar tara de transmisión, siempre que no se exceda el tamaño máximo de un mensaje. Esto se efectúa bajo control del usuario TC, el cual especifica, de forma explícita, cuándo desea el envío de uno o de todos los componentes en espera de transmisión. Dichos componentes son aquellos para los que el usuario TC ha emitido previamente una primitiva de componente de tratamiento con el mismo ID de diálogo.

Hasta tanto la capa de SCCP del SS N.º 7 proporcione una capacidad de segmentación y reagrupación, el usuario TC debe asegurar que no se rebasa el tamaño máximo de los mensajes SS N.º 7.

El ejemplo E3 ilustrado en el cuadro 11 muestra el comienzo de un diálogo con un centro de servicio de red, en el que un conmutador solicita instrucciones (operación 1) y recibe una petición para conectar la llamada a una dirección de destino dada, y una petición para enviar información (por ejemplo, anuncios o un mensaje para su presentación visual) a la parte llamante. Ambos componentes están contenidos en un solo mensaje.

TC-COMIENZO y TC-CONTINUACIÓN son primitivas de transmisión que se describen a continuación en 3.2.

Puede haber una primitiva de transmisión para cada componente (con lo que hay un máximo de un componente en cada mensaje), o menos primitivas de transmisión que componentes, lo que permite la agrupación de componentes dentro de un mensaje. Además, la información contenida en los parámetros de las primitivas de transmisión (por ejemplo, información de direccionamiento), se aplica a todos los componentes incluidos en el mensaje.

En el lado de origen, la primitiva que solicita la transmisión aparece detrás de la primitiva de tratamiento de componente. Esto indica que la transmisión de todos los componentes precedentes se efectuará inmediatamente y evita indicar los componentes específicos que han de transmitirse con una primitiva de transmisión dada, permitiendo el uso de primitivas de transmisión sin ningún componente asociado.

En el lado de destino aparece, en primer término, la primitiva que indica la recepción de componentes transmitidos: contiene información de control necesaria a la TC para la entrega de cada componente (si existe) del mensaje. El parámetro "último componente" indica al usuario TC cuál es el último componente del mensaje. Los componentes se entregan al usuario TC de destino en el mismo orden en que el usuario TC de origen la transfirió a la TC.

Cuadro 11/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (1, Proporcionar-instrucciones, Clase = 1) Petición TC-COMIENZO (Parámetros de control)	
	Indicación TC-COMIENZO (Parámetros de control) Indicación TC-INVOCACIÓN (1, Proporcionar-instrucciones) Petición TC-INVOCACIÓN (2, 1, Conectar-llamada) Petición TC-RESULTADO-L (1, Enviar información) Petición TC-CONTINUACIÓN (Parámetros de control)
Indicación TC-CONTINUACIÓN (Parámetros de control) Indicación TC-INVOCACIÓN (2, 1, Conectar llamada) Indicación TC-RESULTADO-L (1, Enviar información)	
Tiempo	

3.2 Facilidades de tratamiento de diálogo

Cuando dos usuarios TC cooperan en una aplicación se necesita generalmente más de una invocación de operación. Debe identificarse el flujo resultante de componentes de forma que:

1) puedan identificarse las componentes relacionadas con el mismo flujo;

2) puedan identificarse flujos correspondientes a diversas instancias de la misma aplicación y permitirles que discurran en paralelo.

El usuario TC considera cada uno de estos flujos como un diálogo y se identifica mediante un parámetro ID de diálogo correspondiente. La facilidad de tratamiento de diálogo ofrecida para este fin es el diálogo estructurado.

Cuando sólo se necesita un mensaje para completar una aplicación distribuida, puede emplearse el mensaje unidireccional del diálogo no estructurado. El originador no espera un informe de la consecuencia (outcome) de la operación (es decir, sólo puede invocar operaciones de clase 4), pero puede recibir un informe de error de protocolo, si lo hay. Este informe de error de protocolo podrá también ser transportado en un mensaje unidireccional.

3.2.1 Diálogo estructurado

3.2.1.1 Generalidades

La utilización de diálogos permite la coexistencia de varios flujos de componentes independientes entre dos usuarios TC. Se utiliza el parámetro ID de diálogo en las primitivas de tratamiento de operación y tratamiento de transmisión (diálogo) para determinar qué componente o componentes pertenecen a cada diálogo.

En los siguientes ejemplos el parámetro ID de diálogo está representado (por convenio) por el primer parámetro de estas primitivas que comienza con la letra D. Cada usuario TC posee su propia referencia para un diálogo determinado. Aquí se representan las referencias locales (las utilizadas en la interfaz). La TC efectúa la correspondencia de estas referencias locales con referencias de protocolo (denominadas ID de transacción) incluidas en mensajes.

Para el tratamiento de diálogos en circunstancias normales se han definido tres primitivas que indican el comienzo del diálogo (TC-COMIENZO) su continuación (TC-CONTINUACIÓN) o su finalización (TC-FINALIZACIÓN). Puede utilizarse cada una de estas primitivas para solicitar la transmisión de 0, 1 o varios componentes. Estos componentes pueden contener información relativa a una o varias operaciones.

En el cuadro 12 se representa, para el ejemplo E2, una secuencia posible en la que el diálogo comienza con una petición de prueba y finaliza cuando se ha enviado el resultado de la prueba.

Cuadro 12/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (D1, 1, Prueba, Clase = 1) Petición TC-COMIENZO (D1, Dirección)	
	Indicación TC-COMIENZO (D2, Dirección) Indicación TC-INVOCACIÓN (D2, 1, Prueba) Petición TC-INVOCACIÓN (D2, 2, 1, Selección de opción, Clase = 1) Petición TC-CONTINUACIÓN (D2)
Indicación TC-CONTINUACIÓN (D1) Indicación TC-INVOCACIÓN (D1, 2, 1, Selección de opción) Petición TC-RESULTADO-L (D1, 2, Opciones) Petición TC-CONTINUACIÓN (D1)	
	Indicación TC-CONTINUACIÓN (D2) Indicación TC-RESULTADO-L (D2, 2, Opciones) Petición TC-RESULTADO-L (D2, 1, Resultado de la prueba) Petición TC-FINALIZACIÓN (D2)
Indicación TC-FINALIZACIÓN (D1, Normal) Indicación TC-RESULTADO-L (D1, 1, Resultado de la prueba)	
Tiempo	
NOTA – D1 y D2 son referencias locales para el mismo diálogo y están en correspondencia con los ID de transacción que aparecen en los mensajes.	

En los mensajes de un diálogo se permite cualquier agrupación de componentes: la TC no comprueba, por ejemplo, que un mensaje con el que finaliza un diálogo no incluye invocaciones de operación de clase 1.

Se presupone el intercambio dúplex completo de componentes: si un usuario TC desea introducir algunas restricciones, por ejemplo el funcionamiento en un modo síncrono, definido para usuarios ROSE, deberá introducir por sí mismo los procedimientos necesarios.

3.2.1.2 Intercambio de mensajes

La transmisión de mensajes se efectúa con la calidad de servicio correspondiente a los servicios de la capa subyacente. La TC no proporciona mecanismos de control de flujo ni de recuperación tras error.

– La primera primitiva de tratamiento de diálogo de un diálogo deberá indicar el comienzo del mismo (TC-COMIENZO). No podrán enviarse mensajes ulteriores desde el lado en que se

origina el diálogo, hasta que se reciba un mensaje en sentido opuesto, que indica la continuación del diálogo.

– Si un usuario TC trata de enviar un gran número de mensajes en un breve intervalo de tiempo, ningún mecanismo de control del flujo de la TC podrá evitarlo.

– Como opción puede solicitarse la entrega secuencial SCCP de clase 1 indicada por el parámetro calidad de servicio. Obsérvese que esta opción puede no estar disponible de extremo a extremo, en casos de interfuncionamiento con una red que no la proporcione.

3.2.1.3 Finalización del diálogo

La TC no impone ninguna restricción a la capacidad de un usuario TC para solicitar la finalización de un diálogo. Por este motivo, pueden perderse mensajes si en la aplicación no se adoptan precauciones relativas a la finalización del diálogo.

En particular, si el protocolo de aplicación permite a ambos usuarios TC la emisión de primitivas TC-FINALIZACIÓN aproximadamente al mismo tiempo, y si esas primitivas desencadenan la transmisión de componentes, es probable que algunos de esos componentes (si no todos) no se entreguen a sus respectivos usuarios TC de destino.

Corresponde a la aplicación la definición, en caso de necesidad, de sus propias reglas relativas al derecho de terminar un diálogo. La TC no las comprobará.

Todo mensaje recibido para un diálogo terminado se descarta si dicho mensaje solicita la finalización del diálogo, y todo mensaje que no sea una finalización o un aborto provoca el aborto del diálogo en la entidad distante.

Debe observarse que un usuario TC no puede rechazar, por medio de una primitiva de petición TC-U-RECHAZO ningún componente recibido en un mensaje de FINALIZACIÓN. Si es importante para una aplicación poder rechazar cualquier componente recibido o recibir información de los rechazos del extremo distante, todos los componentes deberán ser colocados en el mensaje de COMIENZO inicial o en mensajes de CONTINUACIÓN subsiguientes.

El diálogo se termina bien sea por el método de finalización preconvenida, o mediante el envío de un mensaje de FINALIZACIÓN que no contenga componentes o que contenga componentes de RECHAZO (si son aplicables).

Las diferencias entre estas tres formas de finalización de un diálogo son las siguientes.

Finalización preconvenida

Una aplicación típica es el acceso a una base de datos distribuida en la que el usuario solicitante (usuario TC A) no sabe dónde está ubicada la información que busca. El usuario TC A difunde una petición a cada ubicación en la que podría encontrarse la información solicitada y recibirá eventualmente una respuesta del usuario TC que posee esa información. La finalización preconvenida evita el envío de mensajes desde otro destino con el contenido: "No tengo esa información". Solamente el destino que responde continuará el diálogo (si así se desea); los restantes destinos finalizarán, por convenio, el diálogo localmente. El originador de las preguntas finalizará también de forma local el diálogo con los destinos que no respondan, cuando reciba la respuesta a su pregunta. Obsérvese que se trata de un convenio entre aplicaciones: La TC no comprueba su cumplimiento ni ello se indica en el protocolo TC.

En el ejemplo E4 del cuadro 13 se ilustra esta situación con dos destinos (B1 y B2), iniciándose dos diálogos (D1, D2) y (D3, D4). La información solicitada está disponible en B1, el cual decide continuar el diálogo.

Puede también emplearse la finalización preconvenida cuando un usuario TC desea enviar información y no espera respuesta de ningún tipo.

Cuadro 13/Q.775

<table>
<tr><th>Usuario TC A</th><th>Usuario TC B1</th><th>Usuario TC B2</th></tr>
<tr><td>Petición TC-INVOCACIÓN
(D1, 1, Pregunta)
Petición TC-COMIENZO
(D1, Dirección)
Petición TC-INVOCACIÓN
(D3, 1, Pregunta)
Petición TC-COMIENZO
(D3, Dirección)</td><td>Indicación TC-COMIENZO
(D2, Dirección)
Indicación TC-INVOCACIÓN
(D2, 1, Pregunta)

Petición TC-RESULTADO-L)
(D2, 1, Respuesta)
Petición TC-CONTINUACIÓN
(D2)
.....</td><td rowspan="2">Indicación TC-COMIENZO
(D4, Dirección)
Indicación TC-INVOCACIÓN
(D4, 1, Pregunta)
B2 no posee la información:
Petición TC-FINALIZACIÓN
(D4, local)</td></tr>
<tr><td>Indicación TC-CONTINUACIÓN
(D1)
Indicación TC-RESULTADO-L
(D1, 1, Respuesta)
D1 prosigue
D3 termina localmente
Petición TC-FINALIZACIÓN
(D3, Local)</td><td></td></tr>
<tr><td colspan="3">Tiempo</td></tr>
</table>

Finalización básica

Cuando un usuario TC emite la primitiva de petición TC-FINALIZACIÓN provoca la transmisión de todos los componentes pendientes al extremo distante. La TC no comprueba que todas las invocaciones de operación han recibido una respuesta cuando se solicita la finalización del diálogo, ni se notifica al usuario TC de que algunas invocaciones de operación pendientes no han recibido el resultado final.

En el extremo receptor, se considera terminado el diálogo cuando se han entregado al usuario TC todos los componentes recibidos en el mensaje que indica la finalización.

Ejemplo: el diálogo finaliza cuando en la prueba del ejemplo E1, cuadro 14, se recibe una respuesta.

Cuadro 14/Q.775

Usuario TC A	Usuario TC B
.....	.....
	Petición TC-RESULTADO-NL (D2, 1, P1)
	Petición TC-RESULTADO-NL (D2, 1, P2)
	Petición TC-RESULTADO-L (D2, 1, P3)
	Petición TC-FINALIZACIÓN (D2, normal)
Indicación TC-FINALIZACIÓN (D1)	Finalización del diálogo para B
Indicación TC-RESULTADO-NL (D1, 1, P1)	
Indicación TC-RESULTADO-NL (D1, 1, P2)	
Indicación TC-RESULTADO-L (D1, 1, P3)	
Finalización del diálogo para A	
Tiempo	

Aborto por el usuario TC

La facilidad de aborto permite al usuario TC detener el diálogo en cualquier momento. Un caso típico es el abandono del servicio por parte del usuario. Las diferencias principales entre estos casos y la finalización normal, son las siguientes:

– los componentes cuya transmisión está pendiente no se envían a la entidad par;

– en el momento de producirse el aborto, puede transmitirse la información de usuario TC entre pares y entregarla al usuario TC distante.

La secuencia ilustrada en el cuadro 15 muestra un abandono por el usuario en el ejemplo E2.

Cuadro 15/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (D1, 1, Prueba, Clase = 1)	
Petición TC-COMIENZO (D1, Dirección)	
	Indicación TC-COMIENZO (D2, Dirección)
	Indicación TC-INVOCACIÓN (D2, 1, Prueba)
	Petición TC-INVOCACIÓN (D2, 2, 1, Selección de opción, Clase = 1)
	Petición TC-CONTINUACIÓN (D2)

Cuadro 15/Q.775 (*fin*)

Usuario TC A	Usuario TC B
Indicación TC-CONTINUACIÓN (D1) Indicación TC-INVOCACIÓN (D1, 2, 1, Selección de opción) Petición TC-U-ABORTO (D1, Causa)	
	Indicación TC-U-ABORTO (D2, Causa)
Tiempo	

3.2.1.4 Situaciones anormales relacionadas con el mensaje

Estas situaciones se consideran con independencia de sus efectos en la subcapa componente.

Pérdida del mensaje

La TC no proporciona protección contra la pérdida de mensajes. Se identifican tres casos:

1) Con el mensaje comienza un nuevo diálogo: el diálogo existirá en el lado de origen solamente, no permitiéndose ningún mensaje en ningún sentido. Oportunamente, un mecanismo dependiente de la implementación en el extremo de origen finaliza el diálogo.

2) El mensaje es la continuación de un diálogo existente: no se detecta la pérdida. La TC reaccionará (o no) a la pérdida de los componentes incluidos, según se indica en 2.4.1 anterior.

3) Con el mensaje finaliza un diálogo: la TC reaccionará oportunamente (mediante la expiración de un temporizador, como se describen en 2.4.1) si este mensaje contiene una respuesta a una operación de clase 1. En otro caso, un mecanismo dependiente de la implementación podrá finalizar el diálogo en el extremo de destino.

Duplicación de mensajes

La duplicación de un mensaje COMIENZO hace que se abran dos transacciones, como se indica más adelante: cada una de estas transacciones tiene su propio ID local y el mismo ID de destino. El usuario TC detecta a su debido tiempo que algo es incorrecto, por lo que se abortan ambos diálogos.

La secuencia ilustrada en el cuadro 16 muestra una duplicación del mensaje COMIENZO en el ejemplo E2.

Cuadro 16/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (D1, 1, Prueba, Clase = 1) Petición TC-COMIENZO (D1, Dirección)	
Indicación TC-CONTINUACIÓN (D1) Indicación TC-INVOCACIÓN (D1, 2, 1, Selección de opción)	Indicación TC-COMIENZO (D2, Dirección) Indicación TC-INVOCACIÓN (D2, 1, Prueba) COMIENZO duplicado: Indicación TC-COMIENZO (D3, Dirección) Indicación TC-INVOCACIÓN (D3, 1, Prueba) Respuesta al primer comienzo Petición TC-INVOCACIÓN (D2, 2, 1, Selección de opción, Clase = 1) Petición TC-CONTINUACIÓN (D2) Respuesta al segundo comienzo Petición TC-INVOCACIÓN (D3, 2, 1, Selección de opción, Clase = 1) Petición TC-CONTINUACIÓN (D3)
Indicación TC-CONTINUACIÓN (D1) Indicación TC-INVOCACIÓN (D1, 2, 1, Selección de opción) El usuario TC considera que esta invocación es anormal y puede rechazarla o abortar uno de los diálogos: Petición TC-U-ABORTO (D1, Causa)	
	Indicación TC-U-ABORTO (D3, Causa)
Tiempo	

En ese momento, existe todavía un diálogo (con ID local D2) en el lado del usuario TC B pero no en el lado del usuario TC A. El usuario TC B recibirá una indicación de la TC cuando expire la temporización de la operación 2 del diálogo D2 sin respuesta (indicación TC-L-CANCELACIÓN) y puede entonces decidir abortar D2. Obsérvese que la situación sería más difícil de detectar si el usuario TC B no invocase una operación de clase 1.

La duplicación de un mensaje CONTINUACIÓN no detectada por la TC.

Cuando se duplica un mensaje de FINALIZACIÓN, el segundo mensaje se recibe con un ID que no corresponde a un diálogo activo: la TC reacciona descartando el mensaje duplicado.

Secuenciación incorrecta de mensajes

Cuando los mensajes incorrectamente secuenciados no afectan ni al principio ni al final de un diálogo, la secuenciación incorrecta no es detectada por la TC, lo cual puede provocar una

secuenciación incorrecta de componentes, a la que la TC reaccionaría como se indica en 2.4.3 anterior.

Cuando un mensaje, que indica la continuación de diálogo, llega tras un mensaje que indica el final del mismo diálogo, no será entregado y provocará que la TC aborte el diálogo. El usuario TC detectará probablemente la pérdida al recibir una indicación prematura de finalización del diálogo. Si la aplicación necesita recuperarse de este caso, deberá iniciarse un nuevo diálogo.

Adulteración de mensajes

Al recibir un mensaje adulterado, la TC reacciona como se indica en la Recomendación Q.774.

En el cuadro 17 se indica la secuencia de primitivas cuando la TC decide abortar el diálogo tras recibir un mensaje adulterado en el ejemplo E2.

Cuadro 17/Q.775

Usuario TC A	Usuario TC B
Petición TC-INVOCACIÓN (D1, 1, Prueba, Clase = 1) Petición TC-COMIENZO (D1, Dirección)	
	Indicación TC-COMIENZO (D2, Dirección) Indicación TC-INVOCACIÓN (D2, 1, Prueba) Petición TC-INVOCACIÓN (D2, 2, 1, Selección de opción, Clase = 1) Petición TC-CONTINUACIÓN (D2)
Mensaje adulterado: Indicación TC-P-ABORTO (D1, Causa)	Indicación TC-P-ABORTO (D2, Causa)
Tiempo	

3.2.1.5 Relaciones entre el tratamiento del diálogo y los componentes

A continuación se dan orientaciones sobre cuándo puede solicitarse la finalización del diálogo. Si no se respetan, la TC no rechazará la petición de finalización del diálogo.

Se han examinado anteriormente los problemas que pueden surgir por colisión de mensajes que soliciten la finalización del diálogo.

No se solicitará la finalización normal cuando:

– existan, para el diálogo, invocaciones de operaciones pendientes;
– el protocolo de aplicación anticipe que podrían rechazarse las respuestas transmitidas con la petición de terminación.

Muchas aplicaciones podrían no definir escenarios de recuperación como reacción a una respuesta rechazada. Esto justifica la transmisión de respuestas o de operaciones de clase 4 en un mensaje que indica la finalización de un diálogo. Las demás aplicaciones deberán usar el servicio de red con conexión o finalizar el diálogo con un mensaje que no contenga componentes y que se enviará solamente cuando ya no pueda recibirse una indicación de rechazo.

Se recomienda no enviar en un mensaje FINALIZACIÓN una operación para la que se espera una respuesta (es decir, se recomienda que el usuario TC no emita una primitiva petición TC-FINALIZACIÓN para activar el componente asociado). Cabe señalar que esto no constituye un requisito del protocolo TC ni del usuario TC, sino simplemente una orientación para este último. Si el usuario TC elige enviar un componente de invocación para una operación de clase 1, 2 ó 3 en un mensaje de FINALIZACIÓN, ello no tiene repercusiones perturbadoras para las máquinas de protocolo TC pares o los usuarios TC. La forma en que el lado de realización descarta los componentes generados como resultado de la ejecución de la operación cuando no hay ningún diálogo activo es un asunto local. Evidentemente, el hecho de que el usuario TC que invoca la operación haya decidido incluirla en un mensaje FINALIZACIÓN significa que en este caso ha estimado conveniente invalidar la semántica inherente a la operación y que no le interesa la información sobre si la operación ha podido llevarse a cabo o no.

3.2.1.6 Aspectos de direccionamiento

El usuario TC que inicia un diálogo debe proporcionar al TC la dirección de destino y la dirección de origen. El usuario TC que proporciona la dirección de origen es responsable de presentar dicha dirección de tal forma que el TC distante pueda utilizarla, sin comprobaciones, para alcanzarlo.

Los TC por encima de SCCP sin conexión utilizan cualquiera de las opciones de direccionamiento suministradas por la SCCP. Por lo tanto, se permite cualquier combinación de tipos de dirección de destino y de origen.

Durante el establecimiento de un diálogo, la combinación de direcciones puede ser optimizada tanto por el usuario TC del lado B como por la TC.

El parámetro opcional de dirección de origen de la primera primitiva petición TC-CONTINUACIÓN puede utilizarse para cambiar la dirección que debe emplearse para encaminar los mensajes subsiguientes asociados con el diálogo. El destino real no deberá resultar afectado por esta modificación. Si hay que alcanzar un nuevo destino, se debe terminar el diálogo y comenzar uno nuevo con el nuevo destino.

A continuación se dan ejemplos de cambios de dirección que no modifican el destino real:

– La dirección inicial era un título global, y la dirección subsiguiente es un PC y SSN para el mismo destino a fin de permitir un encaminamiento óptimo (generalmente sin cruzar una frontera de red).
– Se utiliza un título global general para seleccionar una base de datos de un conjunto de bases de datos duplicadas. La base de datos que responde devuelve su propio título global específico.

3.2.1.7 Calidad de servicio

Las primitivas de petición de tratamiento de diálogo permiten al usuario TC solicitar una determinada calidad de servicio a la capa de red. Si el usuario TC no realiza ninguna petición específica, la SCCP selecciona opciones por defecto (es decir, sin opción de devolución, y sin secuenciamiento).

Si se solicita el control de secuencia, el usuario TC también es responsable de proporcionar la información que permita a la capa de red identificar un flujo de mensajes relacionados entre sí que deben entregarse en secuencia.

Se recomienda que los usuarios TC soliciten el control de secuencia cuando utilicen los servicios TC-RESULTADO-NL.

Cuando se solicita la opción de retorno, se notifica al usuario TC mediante la primitiva indicación TC-AVISO que no se entregará el mensaje asociado.

Aunque la indicación TC-AVISO tiene por objetivo básico el ser procesada por una función de gestión (por ejemplo, cuando informa que ocurrió un fallo durante la traducción de un título global), existen casos en los que en la primitiva petición TC-COMIENZO o petición TC-UNI puede solicitarse la opción de retorno para determinar si la entidad de aplicación de destino (identificada mediante el número de subsistema en la dirección de destino) existe en la entidad receptora. En ese caso la indicación TC-AVISO puede ser interpretada en tiempo real.

En el ejemplo siguiente el usuario TC A que reside en una central local inicia un procedimiento de búsqueda hacia delante antes de establecer una llamada. Solicita la opción de retorno al emitir la petición TC-COMIENZO para comenzar el diálogo con la central de destino B. El procedimiento de búsqueda hacia delante no está implementado en el nodo B. Por lo tanto, el número de subsistema correspondiente con el procedimiento de búsqueda hacia delante no existe y la SCCP del nodo B devuelve el mensaje SCCP que contiene el mensaje COMIENZO. El usuario TC A recibe la confirmación a través de la primitiva indicación TC-AVISO y comienza el procedimiento de establecimiento de llamada básica (circuito).

Cuadro 18/Q.775

Usuario TC A		Usuario TC B
Comienza procesamiento de llamada Petición TC-INVOCACIÓN (D1, 1, Búsqueda hacia delante) Petición TC-COMIENZO (D1, opción de retorno, Dest = dirección B, Orig = dirección A)		
	DATOS UNIDAD (opción de retorno, dirección B, dirección A, COMIENZO)	
		SSN-B no existe
	SERVICIO DATOS UNIDAD (causa de retorno, dirección A, dirección B, COMIENZO)	
Indicación N-AVISO (D1, causa de retorno, Dest = dirección A, Orig = dirección B)		

3.2.2 Diálogo no estructurado

Un mensaje unidireccional contendrá únicamente invocaciones de operación de clase 4 o informes de error de protocolo en tales invocaciones. En un mensaje unidireccional pueden transmitirse múltiples componentes, siempre que el mensaje no rebase el tamaño máximo permitido.

3.3 Facilidades de control de diálogo mejorado

3.3.1 Consideraciones generales

A medida que aumente el número de aplicaciones de señalización que utilizan TC, será necesario poder diferenciar entre ellas durante una instancia de comunicación en especial cuando un cierto número de aplicaciones residan en la misma ubicación en un nodo SS N.° 7.

Las funciones opcionales y el protocolo de la porción de diálogo ofrecen la posibilidad de señalar al principio del diálogo cuál es el protocolo de aplicación (entre los muchos posibles) que interviene en el intercambio subsiguiente de mensajes. La porción de diálogo opcional permite la negociación del contexto de aplicación y como una opción más, la transferencia transparente de los datos de usuario que no son componentes. Esta última puede utilizarse para, por ejemplo, datos de inicialización, versiones de protocolos de usuario, mejoras del contexto de aplicación, contraseñas, etc.

3.3.2 Utilización del contexto de aplicación

El usuario TC que inicia un diálogo puede proponer un contexto de aplicación a su par incluyendo un nombre de contexto de aplicación en la primitiva petición TC-COMIENZO. El contexto de aplicación se refiere al conjunto de ASE y a las reglas de coordinación asociadas que pueden necesitarse durante el diálogo.

Si el nombre del contexto de aplicación es aceptable, el usuario TC que responde puede optar por continuar o finalizar el diálogo de forma normal. Excepto si solicita una terminación preacordada, incluirá el mismo nombre AC en la primera (o única) primitiva de petición de tratamiento de diálogo que utilice.

Si el nombre AC no es aceptable, el usuario TC puede elegir entre:

i) descartar los componentes recibidos y emitir una primitiva petición TC-U-ABORTO para indicar que rehusa el diálogo. En esta primitiva incluirá un nombre de contexto de aplicación, ya sea el recibido u otro que deberá ser utilizado por el iniciador del diálogo para realizar una nueva tentativa. TC no proporciona una característica normalizada para permitir al usuario TC proponer más de un nombre de AC alternativo. Sin embargo, tal procedimiento puede definirse fuera del ámbito del TC mediante el parámetro de información de usuario (véase 3.3.3);

ii) continuar el diálogo pero indicar que hace uso de un AC alternativo (por ejemplo, uno que no utilice el o los ASE que no admite) incluyendo otro nombre de contexto de aplicación en la primera primitiva petición TC-CONTINUACIÓN. Los componentes recibidos pueden descartarse o no, según lo acordado previamente entre los usuarios TC;

iii) finalizar el diálogo de forma normal indicando que la respuesta o respuestas incluidas en el mensaje FINALIZACIÓN se basan en un AC alternativo (por ejemplo, uno que no utilice el o los ASE que no admite), mediante la inclusión de otro nombre de contexto de aplicación en la primitiva petición TC-FINALIZACIÓN.

El usuario TC puede proporcionar igualmente un nombre de contexto de aplicación cuando utilice un servicio TC-UNI. En tal caso, el nombre AC indica al usuario TC par cómo debe interpretar los componentes recibidos.

Es importante señalar que la información de contexto de aplicación cursada en las APDU de tratamiento de diálogo es un nombre del tipo IDENTIFICADOR DE OBJETO. Tal nombre es una referencia a una especificación (documento) en la cual figura la descripción del contexto de aplicación. Dicho documento puede hacer referencia a otras especificaciones en las que, por ejemplo, se indique la sintaxis abstracta de algún protocolo de aplicación. Estas especificaciones pueden presentarse en una notación formal o semiformal o en lenguaje claro.

La especificación de los contextos de aplicación, sus semánticas, la asignación de un valor identificador de objeto y la distribución de esta información a todas las partes que desean comunicar constituyen el proceso de registro de un contexto de aplicación. En el caso en que los contextos de aplicación se registran como parte de las Recomendaciones sobre señalización RDSI, un ejemplo de un valor típico puede ser {ccitt recommendation q xxx ac-name(y)}, para el contexto de aplicación $y^{ésimo}$ descrito en la Recomendación q.xxx.

Si bien en principio es posible una descripción de contexto de aplicación muy detallada y añadir nuevos contextos de aplicación para abarcar cualquier situación que pueda aparecer, puede resultar más sencillo mantener tales especificaciones si el número de contextos de aplicación se mantiene dentro de límites razonables. Por ejemplo, supóngase que un nombre de contexto de aplicación se refiere a la utilización combinada de los ASE A y B. En algunas circunstancias, puede ser necesario señalar que sólo se va a utilizar un subconjunto de las capacidades de A y/o B en una determinada instancia de comunicaciones. En vez de registrar un nuevo nombre de contexto de aplicación para contemplar este caso, puede cursarse la misma información en alguna sintaxis mutuamente aceptable en el campo de información de usuario de las APDU, de AARQ y AARE.

3.3.3 Transferencia de los datos de usuario

Las primitivas de tratamiento de diálogo TC permiten al usuario TC solicitar al TC que transfiera información no relacionada con las facilidades de tratamiento del componente (es decir, no basadas en el paradigma de operación a distancia). Esta información se cursa en el campo de información de usuario de las PDU de control de diálogo o directamente en la porción de diálogo, una vez establecido éste.

Una situación típica en que se necesita esta facilidad es en el establecimiento de un diálogo para enviar algunos datos de inicialización a los pares (perfeccionamiento del contexto de aplicación, datos de autenticación, identificación de un subproceso de destino en el usuario TC, etc.).

Además, esta facilidad puede utilizarse para la negociación del contexto de aplicación: cuando el usuario TC rehusa un diálogo (aborto de usuario en el estado pendiente de diálogo con razón del aborto = contexto de aplicación no soportado), puede insertar una lista de nombres de contexto de aplicación alternativos en el campo de datos de usuario de la primitiva petición TC-U-ABORTO. Estos nombres se cursan como parte de los datos de usuario de la unidad de datos de protocolo de diálogo (ABRT). El usuario TC que ha originado la petición de establecimiento de diálogo puede efectuar una nueva tentativa con uno de esos contextos.

Para utilizar esta facilidad, los dos usuarios TC deberán definir la sintaxis y la semántica de la información a cursar, caso de existir, en cada APDU de diálogo para cada contexto de aplicación. Como el tipo ASN.1 de esta información de usuario es EXTERNO, la sintaxis de esta información puede escribirse en ASN.1 o cualquier otra notación específica de usuario. La forma en que se codifica esta información puede ser también específica del usuario. El tipo EXTERNO permite incluir un valor de datos de una sintaxis abstracta (en este caso una sintaxis específica de usuario) dentro de otra (la de las APDU de diálogo).

La Recomendación X.208 define el tipo EXTERNO de la forma siguiente:

```
EXTERNAL ::= [UNIVERSAL 8] IMPLICIT SEQUENCE {
        direct-reference                OBJECT IDENTIFIER OPTIONAL,
        indirect-reference              INTEGER OPTIONAL,
        data-value-descriptor           ObjectDescriptor OPTIONAL,
        encoding                            CHOICE {
                single-ASN1-type                    [0] ANY,
                octet-aligned                       [1] IMPLICIT OCTET STRING,
                arbitrary                           [2] IMPLICIT BIT STRING }}
```

De las tres formas de referencia para identificar el tipo y la codificación del valor de datos contenido en la construcción EXTERNA, los usuarios TC deben utilizar la referencia directa. El nombre de referencia directa proporcionará la clave para identificar la sintaxis abstracta del valor de datos y las reglas de codificación aplicables al mismo. La referencia indirecta se utiliza para identificar el contexto de presentación, cuya utilización no está prevista actualmente en el sistema de señalización N.º 7. Además de la referencia directa, el usuario TC puede proporcionar igualmente

una descripción explícita del valor de datos en una notación informal mediante el empleo del descriptor de valor de datos.

Si el valor de datos externo es un solo tipo ASN.1 y se utilizan las reglas de codificación básicas para codificar este valor, puede utilizarse cualquiera de las posibilidades previstas para el campo "codificación". Si la codificación acordada para este valor de datos da lugar a un número completo de octetos, puede utilizarse la posibilidad de codificación con "alineación de octetos" o "arbitraria". Si la codificación acordada para este valor de datos externo no da lugar a un número completo de octetos, debe utilizarse codificación "arbitraria".

Como el protocolo permite la presencia de una SECUENCIA DE EXTERNO en el campo de información de usuario opcional de las APDU de control de diálogo, los dos usuarios TC no están restringidos, cuando definan un contexto de aplicación, a ningún número en la secuencia. (Si la SCCP no proporciona segmentación, los usuarios TC tendrán que asegurarse de que no se viola la restricción en cuanto al tamaño del mensaje en el sistema de señalización N.° 7.)

3.3.4 Aspectos de retrocompatibilidad

Las nuevas funciones y protocolos descritos en las subcláusulas anteriores son optativos, y la especificación de sus procedimientos de protocolo en las Recomendaciones Q.771 a Q.774 son fácilmente distinguibles y pueden suprimirse sin dificultades en los documentos de adquisición, las especificaciones de implementación y las especificaciones de interfaz entre redes basadas en estas Recomendaciones. En ese caso, quedan los mensajes TC definidos en las Recomendaciones de 1988. Ninguna red está obligada a admitir estas prestaciones si no ofrece servicios que exigen estas capacidades.

Un nodo que soporte la versión 1988 de TC no entenderá las APDU asociadas con el contexto de aplicación generado por un nodo conforme a la presente Recomendación (1992) (o cualquier versión posterior) y, por consiguiente, abortará la transacción utilizando la causa P-ABORTO "parte de transacción incorrecta". Si un usuario TC situado en un nodo que soporta las TC conforme a esta Recomendación (1992) (o cualquier versión posterior) recibe un mensaje de aborto con la causa antes mencionada en respuesta a una iniciación de diálogo basada en una información de contexto de aplicación, debe interpretarlo, al menos en las situaciones en que la red soporta una combinación de implementaciones de TC basadas en esta Recomendación (1992) (o cualquier versión posterior) y en las Recomendaciones de 1988, como el resultado de comunicar con un nodo que soporta únicamente las Recomendaciones sobre TC de 1988, y no como el resultado de un error de sintaxis verdadero (lo que es algo sumamente improbable). Por consiguiente, el usuario TC puede intentar una retransmisión del mensaje sin la información de contexto de aplicación adicional, siempre que, evidentemente, esta información no sea crucial para la aplicación.

Es probable que durante un periodo de tiempo una red soporte tanto implementaciones de TCAP conformes a las Recomendaciones de 1988 como implementaciones conformes a esta Recomendación (1992) (o cualquier versión posterior), así como aplicaciones que exijan o no el mecanismo de contexto de aplicación. El despliegue de tales capacidades cae fuera del ámbito de las normas, pero convendrá tenerlo en cuenta cuando se empiecen a prestar los servicios a fin de evitar la ineficacia que supone transmitir dos veces el mensaje inicial.

4 Directrices para la formulación de especificaciones de protocolos de usuarios TC

4.1 Introducción

La Recomendación Q.1400 describe cómo se estructuran los elementos del servicio de aplicación (ASE, *application service element*), los contextos de aplicación (AC, *application-contexts*) y las

entidades de aplicación (AE, *application entities*) y cómo se direcciona una AE en el sistema de señalización N.° 7. En esta cláusula se explica esa arquitectura a partir de la descomposición funcional de una aplicación y se describe cómo deben definirse AE, AC, ASE, las operaciones y los errores.

4.2 Descomposición de la funcionalidad

4.2.1 Proceso de aplicación y entidad de aplicación

Un proceso de aplicación (AP, *application process*) de señalización se comunica a través de una parte de su soporte lógico dedicada exclusivamente a comunicaciones, denominada entidad de aplicación (AE). Por consiguiente, una AE contiene todas las funciones necesarias para las comunicaciones entre AP distribuidos. Para resumir la subcláusula 4.1/Q.1400, un tipo AE es un conjunto de protocolos de comunicaciones específicos de la aplicación. La definición de un tipo AE es un asunto local. Una AE es la realización del correspondiente tipo AE en la entidad física.

4.2.2 Elemento del servicio de aplicación

En muchas ocasiones se observa que las funciones de comunicaciones de una cierta variedad de aplicaciones pueden agruparse en conjuntos de acciones integrados, de tal forma que cada uno de estos conjuntos puede emplearse en más de una AE. Un conjunto de acciones integrado que puede utilizarse en varias AE recibe el nombre de elemento del servicio de aplicación (ASE). Evidentemente, siempre hay algunas funciones de comunicaciones que son específicas de la aplicación y que pueden utilizarse únicamente para satisfacer las necesidades de comunicaciones de la aplicación para la que han sido definidas.

La TC proporciona un medio genérico para que todas las aplicaciones de señalización comuniquen con arreglo al paradigma de operaciones a distancia por conducto de un servicio de red sin conexión.

Un ASE de usuario TC incluye un conjunto de operaciones a distancia que, junto con la TC proporcionan de forma colectiva algunos protocolos de comunicaciones globales para una aplicación de señalización. La definición del ASE especifica también cuál es el usuario TC par que puede invocar cada una de las operaciones, y en qué orden. Si cualquier usuario TC puede invocar cualquiera de las operaciones, se dice que el ASE es simétrico. En 4.5 se describe la forma de definir y agrupar las operaciones.

Desde la perspectiva del usuario TC, el mecanismo para obtener los servicios de un ASE de usuario TC es la invocación de las operaciones de este ASE. Cada operación proporciona una parte del servicio del ASE de una forma intrínsecamente asimétrica, ya que es invocada por un usuario TC y ejecutada por otro (el usuario par distante). Sin embargo, los usuarios TC no son siempre asimétricos (es decir, uno limitado siempre a llevar a cabo las operaciones y el otro a invocarlas), sino que cada uno de ellos puede ser capaz de invocar o realizar una misma operación o distintas operaciones. [De hecho, la interfaz de servicio, desde el punto de vista del usuario TC (que es objeto de estudios ulteriores), puede tener un aspecto muy distinto del proporcionado por los TC. Por ejemplo, la invocación de una operación de clase 1 puede ser considerada por el usuario TC como la invocación de un servicio confirmado, mientras que, desde la perspectiva de la interfaz del servicio TC, es la consecuencia de dos servicios no confirmados, a saber, TC-INVOCACIÓN y TC-RESULTADO.]

Algunos contextos de aplicación pueden requerir utilizar un ASE de usuario TC específico para el establecimiento y liberación del diálogo. Dicho ASE implica el conocimiento de dos operaciones específicas [conocidas como operaciones de vinculación (*bind*) y desvinculación (*unbind*)] que conforman un paquete de conexión.

Otras aplicaciones pueden requerir también el uso de ASE adicionales que no están basadas en la utilización de operaciones. Las PDU de dichas ASE se transportan en la parte del diálogo.

4.2.3 Comunicación entre pares de AE/ASE

La comunicación entre dos AE de diferentes nodos es posible en tanto en cuanto los dos AE son conscientes de la dirección de su par y soportan un contexto de aplicación común. Un contexto de aplicación es la especificación de las reglas, procedimientos y comportamiento en la interfaz exterior entre dos AE durante una comunicación entre ellas. En 4.3 se definen los medios para especificar un contexto de aplicación.

Para aplicaciones basadas en TC que utilicen las Recomendaciones en versión de 1988 (*Libro Azul*) y que se soporten en el servicio DATOS UNIDAD de la SCCP, el número de subsistema SCCP (SSN, *subsystem number*) proporciona el encaminamiento de mensajes del sistema de señalización N.º 7 hacia el AE ubicado en una entidad física que soporta el protocolo de aplicación específico utilizado y, asimismo, implícitamente la codificación empleada para sus mensajes. Es decir, el SSN además de ser parte de la información de direccionamiento, define implícitamente el contexto de aplicación (a grandes rasgos, el conjunto del mensaje) y el contexto de presentación (la codificación de los mensajes).

Con la introducción del protocolo para la parte de diálogo de la TC, que contiene el mecanismo de negociación del contexto de aplicación, no existe relación directa entre el AE y el SSN. El contexto de aplicación permite una forma explícita de reconocer la granularidad de la capa de aplicación. Debido a que (véase 6.1.3/Q.1400) un AE puede soportar un número indeterminado de ASE, el contexto de aplicación identifica el conjunto adecuado de las ASE que deben utilizarse durante el diálogo del sistema de señalización N.º 7.

Todo ello se ilustra en la figura siguiente.

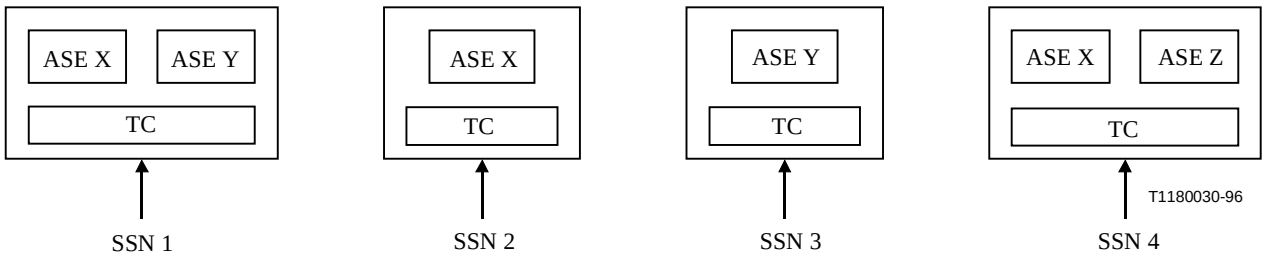


Figura A — Figura B — Figura C — Figura D

Cada una de las cajas muestra un tipo de AE que, a su vez, contiene varios ASE, uno de los cuales, en el caso que es de interés aquí, es siempre la TC. Los X, Y y Z representan varios ASE de usuario TC, tales como los correspondientes a los servicios suplementarios. Cada tipo de AE es accedido por un SSN distinto.

Las figuras A y B muestran dos posibles implementaciones locales de ASE X. En un caso, como muestra la figura A, el ASE X se ha situado junto al ASE Y en un tipo de AE que es accedido por un SSN específico (que posiblemente ha sido seleccionado localmente). En la figura B el ASE X se mantiene independiente. Éstas son dos posibles implementaciones de los servicios proporcionados por el ASE X y la elección de uno u otro no está sujeta a normalización. En la figura D, por ejemplo, se ha elegido otra implementación en la que se agrupa ASE X con otro ASE completamente diferente, el ASE Z, a fin de constituir un tipo de AE al que se accede en SSN 4.

Nótese que los tipos de AE de las figuras anteriores son diferentes para enfatizar que los tipos de AE *no* están normalizados, de forma que cada nodo no tenga que elegir la misma implementación. Las comunicaciones para proporcionar los servicios del ASE X pueden tener lugar entre dos

implementaciones diferentes, una como la de la figura A y otra como la de las figuras B o D, en tanto que los datos de encaminamiento de la SCCP sean los correctos (lo cual es propio de la administración y gestión de red) y que los códigos de operaciones/parámetro/error de ASE X sean distintos de los de los ASE Y y Z (lo cual debe ser decidido en el momento de la especificación; véanse los detalles en 4.5.8).

El único aspecto sujeto a normalización es el comportamiento externo de un sistema. Los contextos de aplicación definen los comportamientos externos visibles entre dos AE que establecen una comunicación. Algunos AC pueden estar normalizados. Los ASE están con frecuencia normalizados para permitir que sean reutilizados en diferentes contextos de aplicación.

4.3 Especificación de un contexto de aplicación

Durante una comunicación, las interacciones entre dos AE así como las interacciones entre los ASE dentro de una AE, están determinadas por las reglas de un contexto de aplicación (AC).

La definición de una AC debe contener al menos:

– una descripción general;

– una definición del protocolo de aplicación completo entre las AE pares mediante:

  i) la identificación de cada ASE utilizado por el AC, indicando cuál de las AE pares inicia el servicio;

  ii) la indicación de las reglas de coordinación entre estos ASE (por ejemplo, concatenación de las PDU procedentes de distintos ASE de usuario TC, cualquier limitación en el orden en el cual pueden invocarse las operaciones de los distintos ASE de usuario TC, etc.) además de las reglas inherentes a las especificaciones de ASE;

  iii) la indicación de las sintaxis abstractas requeridas por los ASE;

– toda restricción especial para asegurar que las AE pares correspondientes a versiones distintas sean compatibles.

A cada contexto de aplicación se asignará un nombre. Tal nombre es un valor del tipo IDENTIFICADOR DE OBJETO, que es cursado (si es necesario) como el valor del elemento de información nombre de contexto de aplicación en la porción de diálogo.

En 5.4 se especifica una clase de objeto de información ASN.1 que puede utilizarse para definir los aspectos estáticos de los contextos de aplicación basados en TC.

4.4 Especificación de un ASE

La especificación de un ASE de usuario TC debe comprender al menos:

– una descripción general de la finalidad del ASE y de sus procedimientos;

– la lista de operaciones soportadas así como una indicación del lado que puede invocar cada operación (o de que ambos lados pueden invocarla);

– las eventuales reglas sobre la secuencia en que pueden invocarse las operaciones;

– la descripción detallada de los procedimientos;

– el modo de interfuncionamiento de las distintas versiones de protocolo;

– la descripción de las interacciones entre el ASE y los TC en términos de primitivas de servicio TC;

– diagramas SDL.

Las clases de objetos de información OPERATION-PACKAGE y CONNECTION-PACKAGE definidas en la Recomendación X.880 pueden utilizarse para especificar los aspectos estáticos de la definición de los ASE que se fundamentan en la utilización de las operaciones.

4.5 Especificación de operaciones y errores

4.5.1 Consideraciones generales

El conjunto de operaciones y errores que constituye una especificación de protocolo de usuario TC puede describirse mediante uno o varios módulos ASN.1. El número de módulos ASN.1 que habrán de utilizarse se deja a criterio del diseñador del protocolo y se examina más adelante en 4.5.5.

Una posible notación utilizada para definir operaciones y errores se basa en la facilidad MACRO ASN.1 definida en la Recomendación X.208. El MACRO OPERACIÓN y el MACRO ERROR son los tipos de datos asociados respectivamente a una operación y a un error. Sin embargo, conforme la notación MACRO va siendo desplazada de la ASN.1, la Recomendación X.880 proporciona una notación alternativa que utiliza dos clases de objetos de información equivalentes. Asimismo explica cómo se puede migrar de la notación MACRO a la notación de clase de objeto.

Cada operación (error) pertenece a un tipo de operación (tipo de error) que se deriva del tipo de OPERACIÓN MACRO (ERROR MACRO).

A cada tipo de operación o tipo de error deberá darse un nombre (una referencia de tipo ASN.1, que comienza con una letra mayúscula).

A cada operación o error debe darse un nombre (un valor de referencia ASN.1 que comienza por una letra minúscula).

La Recomendación X.219 prescribe que los valores para un conjunto de operaciones y errores tiene que ser único dentro de una sintaxis abstracta. Para la TC, esto significa actualmente que tendrán que ser únicos dentro del alcance de un número de subsistema de un grupo de números de subsistema relacionados de un contexto de aplicación.

La definición de tipo puede combinarse con la asignación de valor o efectuarse en dos pasos, como se ilustra en el ejemplo siguiente.

– La especificación de tipo y la asignación de valor están separadas:

```
OperationTypeExample1 ::= OPERATION
ARGUMENT        ParameterType1
RESULT          ResultType1
ERRORS          { error1, error2 }

operationExample1 OperationTypeExample1 ::= localValue 1
```

– La especificación de tipo y la asignación de valor están combinadas:

```
OperationExample1 ::= OPERATION
ARGUMENT        ParameterType1
RESULT          ResultType1
ERRORS          { error1, error2 }
::= localValue 1
```

La mayor o menor conveniencia de combinar las especificaciones de tipo, y por valor, así como la de utilizar un valor global o local, se discute más adelante en 4.5.6.

La notación definida en la Recomendación X.880 no soporta el enfoque de dos pasos. Sin embargo, el código asignado a una operación o a un error puede modificarse fácilmente utilizando la operación

parametrizada predefinida recode {}. En el ejemplo siguiente operationExample2 se define idéntica a operationExample1 (mismo tipo de operación) pero con un valor diferente.

OperationExample2 ::= recode {operationExample1, 2}

4.5.2 Utilización de la notación OPERATION MACRO

4.5.2.1 Utilización del tipo de notación

Un tipo de operación queda totalmente definido como una instancia de un tipo OPERATION MACRO suplementado por un comentario ASN.1 que indica el valor de temporizador asociado.

Las subcláusulas siguientes dan orientación sobre la utilización de las diversas producciones ASN.1 que forman la descripción OPERATION MACRO.

4.5.2.1.1 Especificación del argumento de operación

La siguiente producción ASN.1 indica cómo habrá de especificarse el argumento de una operación:

Parameter ::= ArgKeyword ParameterType | empty
ArgKeyword ::= "PARAMETER" | "ARGUMENT"
ParameterType ::= NamedType | NamedType "OPTIONAL"

Si se proporciona información en la invocación de operación, debe insertarse una de las palabras clave, PARAMETER o ARGUMENT, y tiene que ir seguida del NamedType que corresponde a la estructura de datos a proporcionar; de no ser así, la palabra clave no deberá estar presente en la descripción de operación.

Se admite la retrocompatibilidad de ambas palabras clave con las especificaciones de usuario TC basadas en anteriores versiones de TC. No obstante, se desaconseja el uso de la palabra clave "PARAMETER" para la definición de nuevas aplicaciones.

Aunque el argumento de la operación es siempre un elemento opcional de un componente de invocación, la especificación del ParameterType indica si su presencia es obligatoria u opcional en el momento de la invocación, para completar la ejecución de la operación. En este último caso, la palabra clave "OPTIONAL" va a continuación del tipo ASN.1.

4.5.2.1.2 Especificación de consecuencias (outcomes) positivas

Las siguientes producciones ASN.1 indican cómo especificar una operación que informa éxito:

Result ::= "RESULT" ResultType | empty
ResultType ::= ParameterType | empty

Si hay que devolver información como resultado de la ejecución de una operación con éxito, la palabra clave RESULT tiene que ir seguida del NamedType asociado con la estructura de datos a enviar. Si no hay que proporcionar información, sino que la clase de operación indica que se trata de una operación de informe de éxito, la palabra clave RESULT tiene que estar presente en la descripción de operación, pero se utiliza la alternativa vacío (empty) de la producción de resultado. Si la palabra clave RESULT no se incluye en una descripción de operación, ello indica que no informa un éxito (es decir, operación de clase 2 ó 4).

Aunque el parámetro resultado de la operación es siempre un elemento opcional de un componente de devolución de resultado (*return result*), la especificación del ParameterType indica si su presencia es obligatoria u opcional desde un punto de vista funcional. En este último caso, la palabra clave "OPTIONAL" va a continuación del tipo ASN.1.

4.5.2.1.3 Errores asociados

Las siguientes producciones ASN.1 indican cómo habrán de especificarse operaciones que informen un fallo:

```
Errors ::= "ERRORS" "{ " ErrorNames" }" | empty
ErrorNames ::= ErrorList | empty
ErrorList ::= Error | ErrorList "," Error
Error ::= value (ERROR) | type
```

Si la operación informa fallo, la palabra clave ERRORS deberá incluirse e ir seguida por la lista de errores asociados; en otro caso, esta palabra clave no deberá estar presente. Los errores incluidos en la lista pueden ser referenciados ya sea utilizando una referencia de tipo, o una referencia de valor (es decir, un código de error).

4.5.2.1.4 Especificación de operaciones enlazadas (linked operations)

Las siguientes producciones ASN.1 indican cómo habrán de especificarse operaciones enlazadas:

```
LinkedOperations ::= "LINKED" "{ "LinkedOperationNames "}" | empty
LinkedOperationNames ::= OperationList | empty
OperationList ::= Operation | OperationList "," Operation
Operation ::= value (OPERATION) | type
```

Si la operación es la operación progenitora (parent operation) de un conjunto de operaciones enlazadas, la palabra clave LINKED debe incluirse e ir seguida por la lista de operaciones vástagos (child operations). Las operaciones vástagos incluidas en la lista pueden ser referenciadas ya sea utilizando una referencia de tipo o una referencia de valor (es decir, un código de operación).

4.5.2.2 Utilización de la notación de valor

La notación de valor para operación es o bien la notación para el valor de un elemento de tipo INTEGER o la notación para un elemento de tipo OBJECT IDENTIFIER. Esto depende de si a la operación se le asigna un valor local o un valor global.

4.5.2.3 Especificación de temporizadores

El valor de temporizador asociado con un tipo de operación tiene que indicarse como un comentario ASN.1 frente a la descripción ASN.1 MACRO del tipo de operación.

4.5.3 Utilización de la notación ERROR MACRO

La notación de tipo para un error es la palabra clave ERROR seguida opcionalmente por la palabra clave PARAMETER y el ParameterType asociado a la información que puede enviarse como parámetro de error. La palabra clave PARAMETER no estará presente si no se asocia información con la condición de error.

Aunque el parámetro de error es siempre un elemento opcional de un componente Return Error, la especificación del ParameterType indica si su presencia es obligatoria u opcional desde un punto de vista funcional. En este último caso la palabra clave "OPTIONAL" sigue al tipo ASN.1.

La notación de valor para un error es o bien la notación para el valor de un elemento de tipo INTEGER o la notación para un elemento de tipo OBJECT IDENTIFIER. Esto depende de si a la operación se le asigna un valor local o un valor global.

4.5.4 Uso de la notación CLASS (objeto de información)

La sustitución de la notación MACRO (definida en la Recomendación X.209) con la notación CLASS (objeto de información, definida en las Recomendaciones X.680 a X.683) mantiene la idea

de que las aplicaciones tienen conceptos complejos, con aspectos que deben de ser expresados como estructuras de datos que se transportan mediante los protocolos de comunicación. Cada uno de estos conceptos se clasifica utilizando una plantilla análoga a la definición de MACRO conocida como **clase de objeto de información**. La plantilla para una clase muestra los atributos de los objetos que pertenecen a dicha clase. Las Recomendaciones X.680 a X.683 eliminan la notación MACRO y la sustituyen por la definición CLASS (clase) de objeto de información.

4.5.4.1 OPERATION (objeto de información) CLASS (la clase de operación)

La Recomendación X.880 define la OPERATION (objeto de información) CLASS. En esta subcláusula se presenta una versión en ASN.1 ligeramente modificada en la que se han suprimido los campos que no son relevantes para las aplicaciones basadas en TC.

La notación define la plantilla para una clase de objetos (operaciones a distancia) a las que se les asigna el nombre OPERATION, que consta de diez campos. Cada campo comienza con el símbolo &, que indica propiedad, y va seguido del nombre del campo, que puede empezar por mayúscula o por minúscula. Esta distinción sirve para identificar los tipos de datos que pueden incluirse en los campos cuando se define una instancia de esta clase.

Las palabras que están totalmente en mayúsculas pueden ser palabras clave normalizadas, tales como CLASS y UNIQUE, así como nombres de clases de objetos como OPERATION y ERROR. Si el nombre de un campo comienza por mayúscula, el campo puede contener un tipo ASN.1 arbitrario (por ejemplo, &Argument, véase ASN.1 más abajo), un conjunto de objetos de información (por ejemplo, &Errors) o un conjunto de valores de algún tipo. Si es un conjunto de objetos de información, a continuación sigue el descriptor de la clase de objeto a la que pertenecen. Por otro lado, si nombre de campo comienza por una minúscula (por ejemplo, &returnResult), toma el valor del tipo ASN.1 (por ejemplo, BOOLEAN) de una clase de objeto de información que sigue. Si un campo se marca como OPTIONAL no necesita contener datos cuando se definan las instancias de la clase. La palabra clave UNIQUE seguida de un campo (por ejemplo, &operationCode) significa que el campo es utilizado para identificar instancias de la clase en cuestión y debe de ser único (unique) en un determinado conjunto o colección de dichos objetos.

Finalmente, los diseñadores de la notación ASN.1 han permitido una cierta cantidad limitada de notación definida por el usuario mediante la construcción WITH SYNTAX que se añade a la definición de clase de objeto de información. Ello permite una notación más amigable para la definición de instancias de una clase (CLASS). En la Recomendación X.880 la sintaxis definida por el usuario para las clases (CLASS) OPERATION y ERROR ha sido elegida deliberadamente para que sea similar a la notación MACRO anterior. En 4.5.7 se definen los cambios (pequeños) que se requieren para convertir una operación existente o una definición de error de la definición de la MACRO en una que utilice las definiciones de CLASS de objeto de información.

```
OPERATION ::= CLASS
{
                                    &ArgumentType              OPTIONAL,
                                    &argumentTypeOptional      BOOLEAN OPTIONAL,
                                    &ResultType                OPTIONAL,
                                    &resultTypeOptional        BOOLEAN OPTIONAL,
                                    &returnResult              BOOLEAN DEFAULT TRUE,
                                    &Errors                    ERROR OPTIONAL,
                                    &Linked                    OPERATION OPTIONAL,
                                    &synchronous               BOOLEAN DEFAULT FALSE,
                                    &alwaysReturns             BOOLEAN DEFAULT TRUE,
                                    &operationCode  Code UNIQUE OPTIONAL
}
WITH SYNTAX
{
```

```
                                [ARGUMENT               &ArgumentType [OPTIONAL
                                &argumentTypeOptional]]
                                [RESULT                 &ResultType [OPTIONAL
                                &resultTypeOptional]]
                                [RETURN RESULT          &returnResult]
                                [ERRORS                       &Errors]
                                [LINKED                 &Linked]
                                [SYNCHRONOUS                  &synchronous]
                                [ALWAYS RESPONDS        &alwaysReturns]
                                [CODE                   &operationCode]
}
Code ::= CHOICE
                                {
                                        local           INTEGER,
                                        global          OBJECT IDENTIFIER
                                }
```

4.5.4.1.1 Especificación del argumento de la operación

El campo &Argument es un campo de tipo opcional en el que puede ubicarse un tipo de ASN.1 arbitrario para definir el argumento de la operación distante invocada. El diseñador de la aplicación proporciona el tipo que debe utilizarse cuando se define una operación específica. En la sintaxis definida por el usuario, el tipo del argumento acompañante sigue a la palabra clave ARGUMENT.

El campo &argumentTypeOptional es un campo de valor opcional, que toma el valor TRUE si un argumento definido puede opcionalmente estar ausente desde un punto de vista funcional, o FALSE si debe estar siempre presente. En la sintaxis definida por el usuario, los dos casos se representan por la palabras clave OPTIONAL TRUE (o bien OPTIONAL FALSE) que siguen a la definición de tipo de ARGUMENT.

4.5.4.1.2 Especificación del resultado de la operación

El campo &ResultType es un campo de tipo en el que puede ubicarse un tipo de ASN.1 arbitrario para definir el resultado de la operación distante. El diseñador de la aplicación proporciona el tipo que debe utilizarse para el resultado cuando se define una operación específica. En la sintaxis definida por el usuario, el tipo de resultado acompañante sigue a la palabra clave RESULT.

El campo &resultTypeOptional es un campo de valor opcional, que toma el valor TRUE si un argumento definido puede opcionalmente estar ausente desde un punto de vista funcional, o FALSE si debe estar siempre presente. En la sintaxis definida por el usuario, los dos casos se representan por las palabras clave OPTIONAL TRUE (o bien OPTIONAL FALSE) que siguen a la definición de tipo de RESULT.

4.5.4.1.3 Especificación de resultados positivos

El campo &returnResult es un campo de valor opcional de tipo BOOLEAN que especifica si se ha informado del resultado positivo de una operación distante. En la sintaxis definida por el usuario, dicho campo se reconoce mediante la palabra clave RETURN RESULT. Su ausencia implica que la operación siempre devuelve un resultado.

NOTA – Este campo garantiza el reconocimiento explícito de si una operación informa tras haberse completado con éxito, aunque no se haya definido para ella un tipo de resultado.

4.5.4.1.4 Errores asociados

El campo &Errors es un campo opcional en el que un conjunto de objetos (errores) definidos por la CLASS ERROR (véase 4.5.5) pueden servir para definir un conjunto de errores que pueden ser devueltos si la operación distante fracasa. En la sintaxis definida por el usuario, el conjunto de errores, si están presentes, aparecen entre corchetes "{...}" a continuación de la palabra ERRORS.

4.5.4.1.5 Especificación de las operaciones enlazadas

El campo &Linked es un campo opcional en el que un conjunto de objetos (operaciones) definidos por la CLASS OPERATION pueden servir para definir un conjunto de operaciones que pueden estar enlazadas a una operación en particular que se ha definido. En la sintaxis definida por el usuario, el conjunto de operaciones enlazadas, si están presentes, aparecen entre corchetes "{...}" a continuación de la palabra LINKED.

4.5.4.1.6 Naturaleza síncrona de la operación

El campo &synchronous es un campo de valor opcional, que toma el valor TRUE si la operación es síncrona (es decir, el invocador debe esperar que la operación devuelva una indicación de retorno antes de invocar una nueva por parte del mismo realizador), o FALSE en cualquier otro caso. En la sintaxis definida por el usuario, el código de operación sigue a la palabras clave SYNCHRONOUS.

NOTA – Si la operación es síncrona el campo &alwaysReturns (véase 4.5.4.1) debe ser TRUE.

Las operaciones de usuario TC son asíncronas. Este campo tiene, si no existe palabra clave, el valor por defecto FALSE (es decir, asíncrono); por lo tanto, las aplicaciones de usuario TC actuales y futuras no deben preocuparse por su presencia. Sin embargo, otras operaciones tales como vinculación y desvinculación (véanse 5.2.2.1 y 5.2.2.2) usan explícitamente este campo en su definición.

4.5.4.1.7 Códigos de operación

El campo &operationCode es un campo de valor opcional, que toma un valor entero (un valor único a nivel local) o un OBJECT IDENTIFIER (un valor único global) y que debe ser diferente a los de cualquier otra operación de un conjunto dado de operaciones. En la sintaxis definida por el usuario, el código de operación sigue a la palabras clave CODE.

Si no se define el valor del código de operación, dicha operación no puede invocarse utilizando una PDU de invocación. Un ejemplo de dicha operación lo constituye la operación vinculación (véase 5.2.2.1) que es invocada mediante la PDU de vinculación-invocación definida en 5.2.2.1.2.

4.5.4.1.8 Clases de operación

El campo &alwaysReturns es un campo de valor de tipo BOOLEAN que especifica si la operación devuelve una indicación de retorno. En la sintaxis definida por el usuario se denota mediante el valor verdadero (true) o falso (false) que acompaña a la palabra clave ALWAYS RETURNS. Su ausencia implica que la operación siempre devuelve una indicación de retorno.

Si la operación devuelve siempre una indicación de retorno, ésta es de clase 1, 2 ó 3. El campo &returnResult ayuda a decidir si la operación es de clase 1 ó 3. La presencia o ausencia del campo &Errors reduce la elección anterior a la clase 1 (o a la clase 3).

4.5.4.1.9 Especificación de los temporizadores

La notación no permite especificar los temporizadores salvo como comentarios de la ASN.1.

4.5.5 ERROR (objeto de información) CLASS

La Recomendación X.880 define la ERROR (objeto de información) CLASS. La descripción ASN.1 siguiente presenta una versión ligeramente modificada, eliminando un campo que no es relevante para las aplicaciones basadas en TC.

```
ERROR ::= CLASS
{
        &ParameterType                  OPTIONAL,
        &parameterTypeOptional          BOOLEAN OPTIONAL,
        &errorCode          Code UNIQUE OPTIONAL
}
WITH SYNTAX
{
        [PARAMETER      &ParameterType      [OPTIONAL   &parameterTypeOptional]]
        [CODE           &errorCode]
}
```

4.5.5.1 Especificación de los parámetros que acompañan a un error

El campo &ParameterType es un campo de tipo opcional en el que puede utilizarse un tipo ASN.1 arbitrario para definir el parámetro que acompaña a un informe de error de una invocación de operación. El diseñador de la aplicación proporciona el tipo que debe utilizarse cuando se define un error específico. En la sintaxis definida por el usuario el tipo de parámetro acompañante sigue a la palabra clave PARAMETER.

El campo &parameterTypeOptional es un campo de valor opcional, que toma el valor TRUE si un argumento definido puede opcionalmente estar ausente desde un punto de vista funcional, o FALSE si debe estar siempre presente. En la sintaxis definida por el usuario, los dos casos se representan por la palabras clave OPTIONAL TRUE (o bien OPTIONAL FALSE) que siguen a la definición de tipo de PARAMETER.

4.5.5.2 Códigos de error

El campo &errorCode es un campo de valor opcional, que es un valor entero (un valor único a nivel local) o un OBJECT IDENTIFIER (un valor único global) y debe ser diferente a los de cualquier otro error de un conjunto dado de errores. En la sintaxis definida por el usuario, el código de error sigue a la palabra clave CODE.

Si no se define un valor de código de error, quiere decir que no podrá devolverse una indicación del mismo utilizando una PDU de retorno de error (Return error). Un ejemplo de dicho error lo constituye el error que indica la posibilidad de vinculación (véase 5.2.2.1) que se devuelve utilizando la PDU de vinculación-error (bind-error) definida en 5.2.2.1.2.

4.5.6 Ejemplos de descripciones de operaciones y errores

Esta subcláusula ilustra la parte de especificación de protocolo que trata de las definiciones de operaciones y errores asociados para un ASE de usuario TC simple. La finalidad de operaciones y errores se describe brevemente en forma textual. Seguidamente las operaciones y errores, así como los tipos de datos asociados, se describen formalmente en un módulo ASN.1.

El siguiente ejemplo se basa en un diálogo ficticio de tipo llamada telefónica gratuita (freephone) entre un centro de conmutación y una base de datos de tipo llamada telefónica gratuita.

4.5.6.1 Finalidades de operaciones y errores

4.5.6.1.1 Proporcionar información de encaminamiento

Esta operación la invoca un centro de conmutación para pedir a una entidad distante que proporcione información de encaminamiento para establecer una llamada a un abonado. La información de encaminamiento proporcionada puede ser un número a que se reenvía (forwarded-to number) y puede depender del número de la parte llamante y/o del servicio básico solicitado. En este último caso se invoca la operación vástago getCallingPartyNumber.

4.5.6.1.2 Obtener número de parte llamante

Esta operación es invocada por un elemento de red para pedir a un centro de conmutación que proporcione el número de la parte llamante asociado con un establecimiento de llamada.

4.5.6.1.3 Número llamado no válido

Este error es retornado por un elemento de red para indicar que el número llamado recibido no es conforme con el esquema de numeración soportado.

4.5.6.1.4 Abonado no alcanzable

Este error es retornado por un elemento de red para indicar que en ese momento no hay información de encaminamiento disponible que corresponda al número llamado.

4.5.6.1.5 Llamada prohibida

Este error es retornado por un elemento de red para indicar que la llamada no puede establecerse porque el número llamante está en conflicto con las condiciones de prohibición asociadas al número llamado.

4.5.6.1.6 Número de parte llamante no disponible

Este error lo retorna un centro de conmutación para indicar que no se puede proporcionar el número de la parte llamante.

4.5.6.1.7 Fallo de procesamiento

Este error lo retorna un elemento de red para indicar un fallo de procesamiento.

4.5.6.2 Especificación ASN.1

El siguiente módulo ASN.1 especifica las operaciones y los errores asociados, así como los tipos de datos que corresponden a los elementos de protocolo descritos anteriormente. En este ejemplo la definición tipo de operaciones y errores se combina con la asignación del valor.

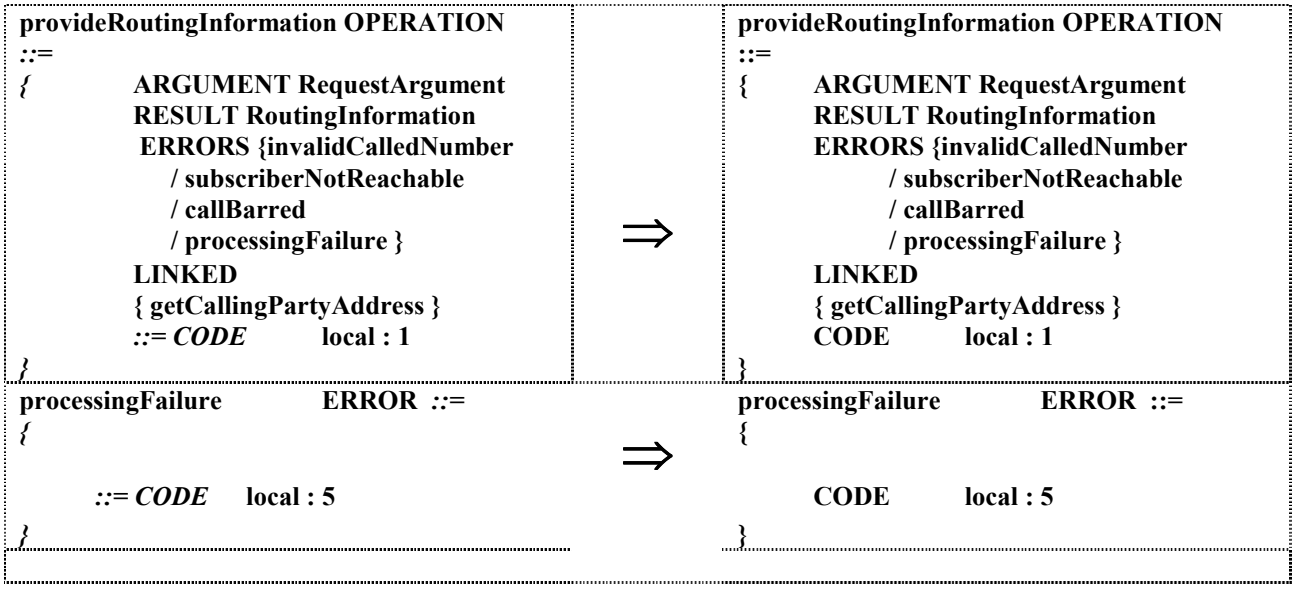
```
TCAP-Examples { ccitt recommendation q 775 modules(2) examples(2) version1(1) }
DEFINITIONS ::=
BEGIN

IMPORTS OPERATION, ERROR
FROM TCAPMessages { ccitt recommendation q 773 modules(2) messages(1) version2(2) };

provideRoutingInformation                OPERATION
ARGUMENT                                 RequestArgument

RESULT                                   RoutingInformation

ERRORS                                   { invalidCalledNumber,
                                         subscriberNotReachable,
                                         callBarred,
                                         processingFailure }

LINKED                                   { getCallingPartyAddress }
-- timer T-pi = 10 s
::= localValue : 1

getCallingPartyAddress                   OPERATION
RESULT                                   CallingPartyAddress
```

```
ERRORS                                  { callingPartyAddressNotAvailable,
                                          processingFailure }

-- timer T-gp = 5 s
::= localValue : 2

invalidCalledNumber ERROR ::= localValue : 1
subscriberNotReachable ERROR ::= localValue : 2
calledBarred ERROR ::= localValue : 3
callingPartyAddressNotAvailable ERROR ::= localValue : 4
processingFailure ERROR ::= localValue : 5
-- data types

RequestArgument  ::= SEQUENCE {
calledNumber            IsdnNumber,
basicService        BasicServiceIndicator OPTIONAL
}

RoutingInformation ::= CHOICE {
        reroutingNumber         [0] IMPLICIT IsdnNumber,
        forwardedToNumber       [1] IMPLICIT IsdnNumber }

BasicServiceIndicator  ::= ENUMERATED {
speech (0),
unrestrictedDigital (1) }

CallingPartyAddress ::= IsdnNumber

IsdnNumber  ::= SEQUENCE {
typeOfAddress           TypeOfAddress,
digits                  TelephonyString }

TypeOfAddress ::= ENUMERATED {
national (0),
international (1),
private (2) }

TelephonyString ::= IA5String  (FROM ("0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"|"*"|"#")) (SIZE (1..15))

END
```

4.5.7 Paso desde la notación MACRO a la notación CLASS (objeto de información)

Todas las especificaciones de usuario TC se han escrito hasta la fecha utilizando la notación MACRO ASN.1 definida en la Recomendación X.208. En esta subcláusula se muestra cómo el uso de estas macros puede transformarse en el uso de la notación CLASS de objeto de información. Tal como se menciona en 4.5.4, la sintaxis definida por el usuario de las clases (CLASS) OPERATION y ERROR es la de la Recomendación X.880 a fin de permitir el mayor grado de características comunes con la construcción anterior. Desde el punto de vista del usuario, hay un pequeño número de cambios algunos de los cuales son sólo un reposicionamiento de varios símbolos mientras que otros permiten expresar en la notación aspectos que con la notación MACRO sólo podían quedar reflejados mediante comentarios.

En esta subcláusula se utilizan ambas notaciones para mostrar la operación *provideRoutingInformation* y el error *processingFailure*, definidos en 4.5.6.2. En ambos casos se muestra a la izquierda la notación macro, de la que deben suprimirse los símbolos subrayados para construir la nueva notación (lado derecho), mientras los símbolos en letra ***cursiva y negrilla*** debe insertarse en la notación anterior para formar la nueva notación.



Utilización de módulos ASN.1

Un módulo es una construcción ASN.1 en la que el diseñador del protocolo reúne varias definiciones de tipos y valores.

Teóricamente, ASN.1 no impone restricciones en cuanto al número de módulos que puedan ser utilizados para definir un protocolo. Todas las definiciones pueden estar contenidas en uno o en muchos módulos. Sin embargo, si las definiciones contenidas en un módulo se necesitan en otro (por ejemplo cuando el error utilizado por una operación se define en otro módulo), la definición correspondiente se hace disponible EXPORTándola desde el módulo en que está definida e IMPORTándola en el módulo en el que se utiliza. Esto se aplica a todos los objetos ASN.1, independientemente de que estén definidos por tipo o por valor.

Ello confiere al diseñador de aplicaciones la libertad de estructurar los módulos de acuerdo con sus necesidades, o según las reglas convencionales que él mismo se imponga. Por ejemplo, un solo módulo podría contener todas las definiciones, particularmente en un entorno con una sola AE y un solo ASE. Otra posibilidad sería que hubiera un módulo para cada definición de ASE, conteniendo cada módulo todas las operaciones y errores utilizados exclusivamente por ese ASE. En el otro extremo, todas las operaciones y errores podrían estar definidos en un módulo de "registro central" y ser exportadas para su utilización en los otros módulos en que estén definidos los ASE.

Puede ser necesario utilizar un modo mixto de notación ASN.1 cuando se definen módulos. Ello surge de la utilización de la notación ASN.1:1994 definida en las Recomendaciones X.680 a X.683 para algunos casos en los que las definiciones se toman de las Recomendaciones UIT-T existentes utilizando la nueva notación. Un ejemplo de ello es la utilización de algunos conceptos de ROSE (Recomendación X.880) tales como las definiciones estáticas de los contextos de aplicación (véase 5.1.1) y la utilización de las operaciones de guía definidas en la Recomendación X.500 para interfaces de red inteligente específicos. Es importante asegurar que los módulos antiguos escritos en ASN.1:1989 no tiene que reescribirse y pueden coexistir con módulos que se han escrito utilizando ASN.1:1994.

En la Recomendación X.680 se proporcionan directrices detalladas para trabajar en este modo mixto que son resumidas a continuación:

- Una especificación puede incluir módulos en ASN.1:1989 y ASN.1:1994. Sin embargo, cualquier módulo DEBE ser conforme a la notación de 1989 o de 1994, pudiéndose incluir comentarios para indicar la versión que se utiliza en cada módulo.

- Las referencias de tipos y valores pueden importarse a un módulo ASN.1:1994 desde un módulo ASN.1:1989 en tanto que:
  a) las definiciones MACRO ASN.1:1989 NO se importan a un módulo que utiliza notación ASN-1:1994; y
  b) están presentes los identificadores para los valores SET, SEQUENCE y CHOICE.
- Las referencias de tipos y valores pueden importarse a un módulo ASN.1:1994 desde un módulo ASN.1:1989 en tanto que:
  a) NO se IMPORTAN nuevos tipos ASN.1:1994 (CHARACTER STRING, UniversalString, BMPString, EMBEDDED-PDV).

4.5.8 Asignación y gestión de los códigos de operación y de error

4.5.8.1 Consideraciones generales

En 4.1 a 4.5.4 se describe cómo pueden especificarse AC, ASE, operaciones y errores. Se expone también cómo los ASE utilizan operaciones y errores y cómo se emplean los propios ASE para definir el contexto de aplicación entre dos AE pares.

Las AC y los ASE constituyen unos instrumentos de modelado y especificación muy útiles para el diseño de protocolos de aplicación. En última instancia, durante un diálogo entre dos usuarios TC, todo protocolo de aplicación basado en la TC está constituido por un intercambio de valores de datos de tipos de operaciones y de errores, identificados respectivamente por sus códigos de operación o error. El único requisito del sistema de señalización N.° 7 (y de ROSE) es que los códigos de operación y de error sean unívocos dentro de la sintaxis abstracta. Como actualmente no hay forma de señalar de manera explícita la sintaxis abstracta a la que pertenece un determinado código de error o de operación, el diseñador de la aplicación debe asegurarse de que estos códigos son únicos en el ámbito de un número de subsistema o de un contexto de aplicación. Cuando el ámbito de los códigos de operación y error sea un contexto de aplicación, el diseñador de la aplicación deberá asegurar igualmente que el nombre del contexto de aplicación se curse al extremo distante mediante el protocolo de la porción de diálogo.

Hay muchos métodos posibles por lo que respecta a la asignación y gestión de los códigos de operación y de error, y es preciso considerar muchos factores. Dos factores muy importantes son la estructura de AC/ASE y la reutilización de operaciones y errores.

4.5.8.2 Importación y exportación de operaciones y errores

Como cualquier otro tipo ASN.1, Operaciones (Operations) y errores (Errors) pueden ser exportados e importados entre módulos ASN.1. Este método puede utilizarse cuando sea necesario definir una operación cuyo tipo corresponde a una operación existente, pero el valor que deba asignarse a esta nueva operación es diferente del asignado a la operación existente (es decir, para fines de unicidad). Esto se ilustra por el siguiente ejemplo en el cual objectIdentifier1 y objectIdentifier2 son identificadores ficticios.

```
ExportingModule { objectIdentifier1 }  DEFINITIONS ::=
BEGIN
EXPORTS operation1, OperationTypeA, error1, ErrorTypeA;

IMPORTS OPERATION, ERROR FROM TCAPMessages
{ ccitt recommendation q 773 modules(2) messages(1) version2(2) };

operation1                  OPERATION
ARGUMENT                    ParameterType1
RESULT                      ResultType1
```

```
ERRORS                    { error1 }
::= localValue 1

OperationTypeA ::=        OPERATION
ARGUMENT                  ParameterTypeA
RESULT                    ResultTypeA
ERRORS                    { ErrorTypeA}

operation2 OperationTypeA ::= localValue : 2

error1  ERROR
PARAMETER DiagnosticType1
::= localValue : 1

ErrorTypeA ::= ERROR
PARAMETER DiagnosticTypeA

error2 ErrorTypeA ::= localValue : 2
-- Note that ParameterType1, ResultType1, ParameterTypeA, ResultTypeA,
-- DiagnosticType1 and DiagnosticTypeA have to be defined somewhere if they are not defined
-- within this module, they have to be imported from the module where they are defined

END

ImportingModule { objectIdentifier2 }  DEFINITIONS ::=
BEGIN
IMPORTS OPERATION, ERROR FROM TCAPMessages
{ ccitt recommendation q 773 modules(2) messages(1) version2(2) };
operation1, OperationTypeA, error1, ErrorTypeA
FROM ExportingModule { objectIdentifier1 };

operation2 OPERATION
ARGUMENT ParameterTypeX – to be defined somewhere in the module
::= localValue : 2

error2 ERROR ::= localValue : 2
-- value 2 is already in use. This value 3 is allocated to the imported objects
operationA OperationTypeA ::= localValue : 3
errorA ErrorTypeA ::= localValue : 3

END
```

4.5.8.3 Influencia de la estructura de ASE/AC en la administración de los códigos de operación y de error

Con respecto a la estructura de AC/ASE, las opciones son las siguientes:

4.5.8.3.1 Enfoque monolítico – Una AC, un ASE

Conceptualmente, éste es el enfoque más sencillo. El protocolo de aplicación se define por una AC que comprende un solo ASE (además de TC). Todas las operaciones utilizadas en ese ASE podrían estar definidas en un solo módulo ASN.1, que contuviera también las definiciones del paquete de operación del ASE y del AC. Dentro del protocolo, todas las operaciones y errores se identifican unívocamente al asignárseles un valor (entero) local único.

La ventaja de este esquema es su sencillez, y su inconveniente radica en que no permite identificar independientemente los bloques de construcción que puedan evolucionar separadamente en el seno de la definición de AC.

4.5.8.3.2 Una AC que comprende más de un ASE

Al definir un protocolo de aplicación, el diseñador puede optar por estructurar una AC de modo que comprenda dos o más ASE. Por ejemplo, puede agrupar los elementos de protocolo referentes a la autenticación de usuario en un ASE separado (que podría ser reutilizado en otro protocolo), y los concernientes a la consulta de bases de datos propiamente dicha en otro ASE. Esto puede facilitar el diseño de un sistema modular pero, cuando se combinen todos los ASE constitutivos para formar la AC, habrá que tener cuidado de asegurarse de que a las operaciones y errores distintas contenidos en ASE diferentes no se les haya asignado el mismo valor.

La utilización de una misma operación/error en dos ASE diferentes de una misma AC no causa ningún problema. Si los valores asignados a esa operación son los mismos en cada caso, dentro del protocolo habrá solamente una operación/error asociado con ese valor. Si se asignan valores diferentes, aunque en el protocolo aparecerá como si hubiese dos operaciones/errores diferentes, en la implementación de la aplicación estos dos valores distintos provocarán la invocación/identificación de la misma operación/error.

Sin embargo, si dentro de una misma AC se asigna a una operación definida en un ASE el mismo valor que tiene asignada una operación diferente en otro ASE, es evidente que esto causará un problema. Cuando un ASE se utiliza en una sola AC, un simple esquema de asignación de código puede evitar este problema. En cambio, cuando se utiliza el mismo ASE en varias AC, la situación será difícil de controlar y las únicas soluciones "seguras" serán las descritas en los apartados i) a iii) siguientes:

i) Dos o más protocolos ASE comparten valores locales comunes de operación/error

Cuando se definen los ASE, el diseñador o diseñadores del protocolo asignan los valores de tal forma que no pueda producirse ningún conflicto. Ello requiere una coordinación en las tareas de definición del ASE, y significa que los ASE comparten la misma sintaxis abstracta.

Un inconveniente de este esquema es que si se utiliza uno de los ASE en más de un contexto (es decir, junto con un conjunto distinto de ASE), es casi imposible evitar conflictos de valores en todas las combinaciones posibles.

ii) Asignación de valores globales (identificadores de objeto) a operaciones y errores

Como un identificador de objeto es único en todo el sistema de señalización N.° 7, no hay peligro de conflicto de valores cuando se combina un ASE con cualquier otro.

Este esquema tiene el inconveniente de que, cuando un identificador de objeto está codificado, es más largo que un entero simple.

iii) Compartición de operaciones/errores asignando tipos cuando se definen operaciones/errores en vez de valores

Esta solución supone que la definición del tipo de las operaciones y errores es independiente de la asignación de los valores.

Cuando un diseñador del protocolo define un contexto de aplicación, recopila todos los tipos de operación y error utilizados por los ASE requeridos y les asigna valores adecuados, de forma que no se produzcan conflictos.

Haciendo esto puede considerarse que el diseñador del protocolo define un nuevo conjunto de ASE que son isomórficos con respecto a los existentes, y que difieren únicamente por los valores de sus operaciones y errores.

4.5.8.4 Reutilización de operaciones y errores

Independientemente del número de ASE incluidos en un protocolo, existen situaciones en que conviene incluir una operación o error existente cuando se define un nuevo ASE.

La operación o el error pueden reutilizarse de una de las siguientes formas.

La operación se importa en uno de los módulos que definen uno de los ASE. Esto sólo es posible si se asegura que no haya conflictos de valores.

Esto puede lograrse si:

i) Existe un registro central de operaciones y errores que utiliza únicamente valores de una gama reservada que nunca es empleada por las operaciones específicas del ASE. Este enfoque impone una limitación en cuanto a los ASE de usuario TC, que tal vez no se satisfaga en un entorno más amplio (es decir, si van a utilizarse operaciones o errores de protocolos ISO o DSS 1).

ii) Se adjudican valores globales a las operaciones y errores. El inconveniente de esto es que un valor global exige codificar más octetos que uno local y necesita igualmente un registro oficial en el árbol de identificador de objeto.

iii) El tipo de operación o de error se importa en uno de los módulos que definen uno de los ASE, donde se ha asignado un valor adecuado. Ello supone que el protocolo de exportación utiliza el método de dos pasos para la definición de operaciones y errores, o que los tipos de operaciones y de errores necesarios están incluidos en un registro central.

iv) La operación o el error están completamente redefinidos. No obstante, puede importarse parte de la definición original (por ejemplo, el tipo del argumento).

4.6 Especificaciones de tipos de datos

4.6.1 Generalidades

Como se ha indicado anteriormente, el tipo de información que puede acompañar a una invocación de operación, el informe de un éxito o el informe de un fallo, se especifica como un tipo de datos ASN.1. Esto también es válido para la información que puede intercambiarse como datos de usuario de la porción de diálogo.

Este tipo de datos puede ser un tipo incorporado (por ejemplo, tipo entero, tipo booleano, tipo nulo, tipo de cadena de octetos, etc.) o estructurado (por ejemplo, tipo de secuencia, secuencia de tipo, tipo de elección, etc.). Puede también derivarse de estos tipos mediante subtipificación (por ejemplo, restricción del tamaño, gama de valores) o rotulado.

4.6.2 Utilización de rótulos

ASN.1 proporciona un mecanismo de rotulación que permite definir un tipo isomórfico con respecto a otro existente, que por ende sólo se diferencia de éste por su rótulo.

Como se indica claramente en la Recomendación X.208, los rótulos (ASN.1) están destinados a ser utilizados exclusivamente por máquinas, fundamentalmente para facilitar el proceso de decodificación.

Los rótulos no están destinados a ser utilizados para la identificación directa de los elementos de información, según éstos se perciben desde el punto de vista de un proceso de aplicación local. La forma en que se identifican localmente estos elementos de información es una cuestión de implementación y depende del diseño del soporte lógico y del lenguaje utilizado para manipular la representación interna de los datos. A este respecto cabe señalar que se requieren rótulos distintos principalmente en las siguientes situaciones:

– los elementos de información son miembros de un conjunto (no ordenado, es decir, un tipo de conjunto) y por consiguiente su posición relativa no puede utilizarse para discriminar entre dos elementos de información del mismo tipo (y por consiguiente con el mismo rótulo);

– los elementos de información son miembros de un conjunto ordenado (es decir, un tipo de secuencia) pero la presencia o ausencia de elementos opcionales hace imposible discriminar entre la presencia de un elemento opcional y la presencia de un elemento de información del mismo tipo inmediatamente siguiente;
– dos apariciones de un mismo tipo base en un tipo de elección.

Hay cuatro clases de rótulos. Además de la clase universal que se utiliza para identificar un tipo incorporado, se definen tres clases para permitir la definición de tipos isomórficos con fines de decodificación:

– La *clase APPLICATION-WIDE* – Los rótulos asignados en esta clase pueden utilizarse para identificar directamente la estructura del tipo de datos que habrán de ser decodificados. Los rótulos asignados en esta clase son significativos a través de una aplicación y no se utilizarán cuando haya riesgo de conflicto entre valores. La clase APPLICATION-WIDE debe utilizarse solamente si la aplicación es un dominio "cerrado" o si hay un registro común (common registry).
– La *clase CONTEXT-SPECIFIC* – Los rótulos asignados en esta clase sólo son significativos en un dominio definido. Por lo tanto, el proceso de decodificación identifica la estructura de datos a decodificarse tanto a partir del valor del rótulo como del contexto en que aparece. Existe el entendimiento común de considerar el contexto limitado a la construcción inmediatamente superior.
– La *clase PRIVATE*, que es muy similar a la clase APPLICATION-WIDE pero está fuera del ámbito de la normalización.

Debe señalarse que la clase CONTEXT-SPECIFIC es la única (cuando se utiliza correctamente) que asegura que nunca habrá conflicto entre valores, cuando se efectúa importación/exportación de tipos de datos entre módulos.

4.6.3 Instancias y tipos

Es necesario distinguir claramente entre un tipo de datos y una instancia (instance) de un tipo de datos, que es la representación abstracta de un elemento de información transportado en una unidad de datos. Para facilitar la especificación y acrecentar la legibilidad, ASN.1 proporciona una notación NamedType que permite calificar una instancia específica de un tipo de datos utilizando un identificador ASN.1.

Debe observarse que no es necesario definir un tipo de datos por cada elemento de información. Cuando dos elementos de información son sintácticamente equivalentes es evidentemente más convenientemente representarlos como dos instancias del mismo tipo de datos, o, si se requiere para la decodificación, como instancias de dos tipos derivados del mismo tipo de datos por una rotulación CONTEXT-SPECIFIC y cuyas definiciones aparecerán entonces solamente dentro de la definición de la construcción superior (es decir, los rotulados sólo se definen en el contexto específico de la construcción superior).

4.6.4 Exportación e importación de elementos de información

Es posible que los protocolos de señalización basados en la TC tengan que utilizar elementos de información definidos en otras especificaciones de protocolo de señalización. Más bien que definir un nuevo elemento de información, deberá preferirse utilizar un mecanismo de importación. Los tipos de datos pueden ser importados formalmente o informalmente, lo que depende de la manera en que está especificado el protocolo exportante.

– El protocolo exportante se especifica utilizando un módulo ASN.1 que exporta los tipos de datos requeridos: los tipos de datos requeridos pueden ser importados formalmente en el módulo donde hay que definir el nuevo protocolo.

– El protocolo exportante no se especifica utilizando módulos ASN.1: un modo conveniente consiste en definir un nuevo tipo de datos isomórfico al tipo cadena de octetos y especificar informalmente una referencia a la especificación en que se define la estructura interna (es decir, utilizando un enunciado de comentario).

4.7 Especificación de sintaxis abstractas

Las especificaciones de ASE y AC suponen una referencia a una o varias sintaxis abstractas. Cada una de ellas representa, en un nivel abstracto (es decir, independiente de las técnicas de codificación), conjuntos de valores de datos que pueden intercambiarse durante la comunicación.

Actualmente no es necesario asignar explícitamente un nombre a la sintaxis abstracta constituida por los mensajes TC para una aplicación determinada, puesto que esta sintaxis abstracta viene identificada implícitamente por el número del subsistema que direcciona la AE. Sin embargo, la estructura de la información de usuario cursada en la porción de diálogo se deberá definir como parte de una o varias otras sintaxis abstractas.

Por consiguiente, el diseñador de protocolo que desee informaciones de usuario que no sean componentes cursados por las TC, definirá en primer lugar una o varias sintaxis abstractas que engloben todos los tipos de datos cuyos valores puedan ser cursados.

Asignará igualmente un nombre a cada una de estas sintaxis abstractas. Dicho nombre, que es un valor del tipo IDENTIFICADOR DE OBJETO, servirá de referencia directa cuando se curse el valor real como parte de un valor construido de tipo EXTERNO, como se especifica en la Recomendación Q.773.

Actualmente no hay un método formal para especificar una sintaxis abstracta; sin embargo, cuando esta sintaxis puede describirse en ASN.1, la manera más sencilla consiste en definir un tipo de elección construido a partir de todos los tipos de datos que constituyen la sintaxis abstracta.

Por consiguiente una sintaxis abstracta puede definirse informalmente incluyendo la frase siguiente en las especificaciones del protocolo:

"El conjunto de valores de datos del tipo Module-X.Type-A forma una sintaxis abstracta identificada por el siguiente nombre de sintaxis abstracta: <objectIdentifierValue>".

En la frase anterior, Type-A es el nombre del tipo de elección, y Module-X es el nombre del módulo en el que se le define.

En este contexto, el nombre de la sintaxis abstracta se refiere también implícitamente a las reglas de codificación que deben aplicarse a la sintaxis abstracta. Tales reglas de codificación, que pueden ser las definidas en la Recomendación X.209 (pero no necesariamente), deben ser acordadas *a priori* entre los usuarios TC.

La clase de objeto de información ASN.1 ABSTRACT-SYNTAX especificada en la Recomendación X.681 puede utilizarse también para definir sintaxis abstractas. El siguiente ejemplo ilustra dicha utilización, definiendo una sintaxis abstracta que incluye los valores del tipo InitData que es un conjunto de tres unidades de datos de protocolo utilizados en el establecimiento del diálogo para transferir una lista de unidades funcionales soportadas o información de autentificación:

```
InitModule DEFINITIONS ::=
BEGIN

InitData ::= CHOICE {
functionalUnits    [0] IMPLICIT FunctionalUnits,
authenticationInfo  [1] IMPLICIT AuthenticationInfo }
```

```
FunctionalUnits ::= SEQUENCE OF FunctionalUnit

FunctionalUnit ::= ENUMERATED {unit(1), unit2(2), unit3(3) }

AuthenticationInfo ::= SEQUENCE {
        algorithm OBJECT IDENTIFIER,
        signature OCTET STRING }
init-abstract-syntax ABSTRACT-SYNTAX ::=
{
InitData            IDENTIFIED BY          { -- some object identifier value -- }
}

END
```

4.8 Reglas de codificación

La sintaxis concreta de mensajes TC (es decir, el tren de bits intercambiado entre TC pares como datos de usuario de mensajes SCCP) se obtiene aplicando las reglas básicas de codificación a la descripción de la sintaxis abstracta de mensajes TC [incluidos los elementos de usuario TC, excepto los cursados como valor de un tipo EXTERNO (por ejemplo, el campo de información de usuario de una APDU de control de diálogo)]. Las reglas básicas de codificación se definen en la Recomendación X.209, y en la Recomendación Q.773 se indican algunas restricciones menores relacionadas con la codificación de la porción TC.

La información de usuario cursada como valor de un tipo EXTERNO puede también codificarse de acuerdo con las reglas básicas de codificación (pero no necesariamente). En ese caso, el correspondiente nombre de la sintaxis abstracta sirve también como referencia implícita a las reglas de codificación aplicadas (véase 3.3.3).

Cabe señalar que las reglas básicas de codificación presentan varias opciones, especialmente para la codificación de longitudes. Esto significa que una implementación deberá poder decodificar una unidad de datos independientemente de las condiciones de codificación seleccionadas por la entidad emisora.

5 Correspondencia entre los conceptos genéricos de las operaciones a distancia (ROS) y los servicios TC

5.1 Visión general

La Recomendación X.880 define un modelo genérico para las comunicaciones interactivas entre objetos, en las que la interacción básica implica la invocación de una operación por parte de un objeto (el invocador) y su realización por parte de otro (el realizador). Este modelo, conocido como de operaciones a distancia (ROS) viene acompañado de un conjunto de objetos de clases de información que deben ser utilizados por los diseñadores de protocolo en la especificación de las aplicaciones basadas en ROS.

La Recomendación X.880 reconoce que existen múltiples realizaciones posibles de este modelo, en lo que a las comunicaciones se refiere. El objetivo de esta cláusula es mostrar cómo y porqué las TC deben considerarse una de dichas realizaciones, proporcionando una correspondencia entre los conceptos genéricos y los servicios TC.

5.1.1 Notación y concepto para el modelo genérico ROS

La Recomendación X.880 define varias clases de objetos de información que son útiles en la especificación de protocolos de aplicación basados en ROS. Dichas clases de objetos se definen

utilizando la notación ASN.1 para la especificación de objetos de información definida en la Recomendación X.681.

La clase OPERATION se utiliza para definir una operación. Es equivalente a la OPERATION MACRO definida en las Recomendaciones X.219 y Q.773. Los diseñadores de aplicaciones usuario TC pueden utilizar esta clase como alternativa a la notación MACRO descrita en la cláusula 4. En el anexo C/X.880 se proporcionan directrices para migrar de la notación MACRO a esta notación.

La clase ERROR se utiliza para definir una operación. Es equivalente a la ERROR MACRO definida en las Recomendaciones X.219 y Q.773. Los diseñadores de aplicaciones usuario TC pueden utilizar esta clase como alternativa a la notación MACRO descrita en la cláusula 4. En el anexo C/X.880 se proporcionan directrices para migrar de la notación MACRO a esta notación.

La clase OPERATION-PACKAGE se utiliza para definir un conjunto de operaciones que sólo pueden ser invocadas por un objeto ROS que asume el papel de "consumidor"(consumer), un conjunto de operaciones que sólo pueden ser invocadas por un objeto ROS que asume el papel de "suministrador" (supplier) y un conjunto de operaciones que pueden ser invocadas por ambos tipos de objetos ROS. Cuando se utilizan servicios de comunicaciones del SS N.º 7 o de OSI, un paquete de operación se implementa como un elemento de servicio de aplicación (ASE).

La clase CONNECTION-PACKAGE se utiliza para definir las operaciones de vinculación y desvinculación utilizadas como parte del establecimiento y liberación de una asociación. Cuando el paquete de conexión se realiza utilizando los servicios de comunicación del SS N.º 7, se implementa como los procedimientos que hacen uso de los servicios de tratamiento de diálogo estructurado de las capacidades de transacción. Puede considerarse que los contextos de aplicación que no requieren la invocación explícita de operaciones de vinculación y desvinculación incluyen un paquete de conexión que utiliza las operaciones predefinidas emptyBind y emptyUnbind.

La clase CONTRACT se utiliza para definir un contrato de asociación en términos de un paquete de conexión y de uno o más paquetes de operación. Cuando se especifica el contrato se identifican aquellos paquetes en los que solo el iniciador de la asociación asume el papel de consumidor, los que sólo el respondedor de la asociación asume el papel de consumidor y aquellos en los que cualquiera de los dos asume el papel de consumidor. Cuando se utilizan los servicios de comunicación del SS N.º 7 o de la OSI, un contrato se establece como un contexto de aplicación.

La clase ROS-OBJECT-CLASS se utiliza para definir un conjunto de capacidades comunes de un conjunto de objetos ROS en términos de los contratos (de asociación) que soportan como iniciadores y/o respondedores. Cuando se realiza utilizando TC u OSI, un objeto ROS se corresponde en una parte de un proceso de aplicación.

Estas clases proporcionan una notación que está disponible para el diseño de aplicaciones basadas en ROS, con independencia de cualquier realización en particular. La especificación de protocolo requiere la definición de un contexto de aplicación que indica como se realiza el contrato de operación. En 5.4 se define una clase de objeto de información APPLICATION-CONTEXT que está disponible para la especificación de realizaciones de un contrato de operación basadas en TC.

5.1.2 Modelo de comunicación

La realización de ROS implica la selección de un medio adecuado que transporte invocaciones y respuestas entre una pareja de objetos ROS.

Los posibles medios pueden clasificarse en dos grandes categorías:

- aquéllas que son requeridas cuando el invocador y el realizador deben implementarse en un mismo equipo físico;

- aquéllas que son requeridas cuando el invocador y el realizador deben implementarse en equipos físicamente separados.

Dentro de la primera categoría pueden considerarse facilidades de paso de mensaje (message-passing) y de llamada de procedimiento (procedure calling).

El medio de la segunda categoría depende del tipo de red que interconecta los dos objetos y de algunos criterios de calidad de servicio.

La Recomendación X.880 modela el medio como compuesto de dos sub-objetos (uno para el invocador y otro para el realizador) y un objeto de transferencia de información (véase la figura 1). Las capacidades del objeto de transferencia de información incluyen también las funcionalidades de control de asociación que pueden ser requeridas para establecer una asociación entre las entidades de aplicación involucradas en la comunicación.

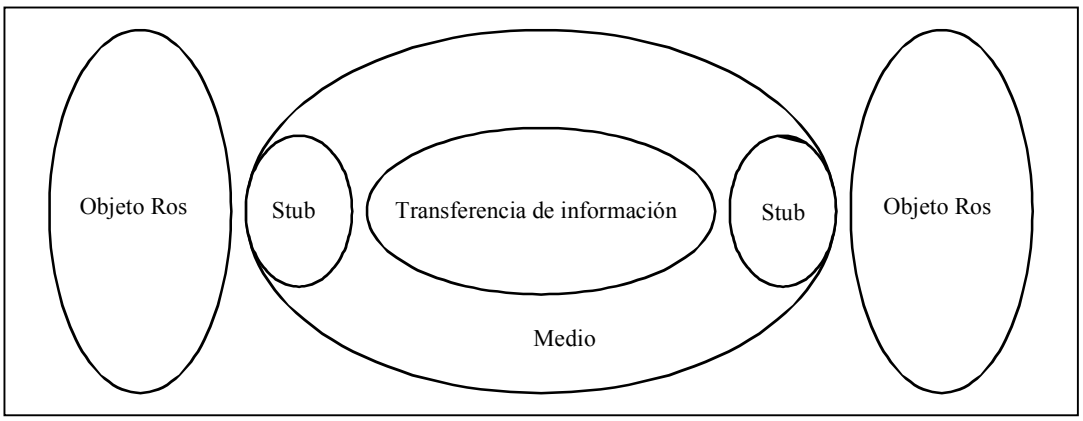


Figura 1/Q.775 – Modelo de comunicación genérico de operaciones a distancia (ROS)

El papel de cada objeto stub es transformar las invocaciones y respuestas en unidades de datos de protocolo (y viceversa) que intercambian utilizando el objeto de transferencia de información. Para un determinado tipo de objetos stub existen varios posibles tipos de objetos de transferencia de información.

En el contexto de la OSI, el objeto stub se implementa mediante el elemento de servicio de operación a distancia (ROSE, *remote operation service element*) estando disponibles varias realizaciones de transferencia de información, y utilizando las combinaciones adecuadas de ACSE, RTSE y el servicio de presentación.

Los bloques de tratamiento de componentes de la subcapa de componente TC, unidos a un conjunto de ASE específicos de operación (los ASE de usuario TC), realizan los objetos stub (véase la Recomendación Q.774). Los CHA cuyos servicios se definen en 3.1.3/Q.771 dirigen el protocolo genérico requerido para invocar e informar de devoluciones de operaciones arbitrarias.

Cada ASE de usuario TC conoce las definiciones de las operaciones específicas involucradas en algunos de los paquetes de operación. Conjuntamente, el CSL y los ASE de usuario TC conocen todas las operaciones de los contratos de asociación. Véase la figura 2.

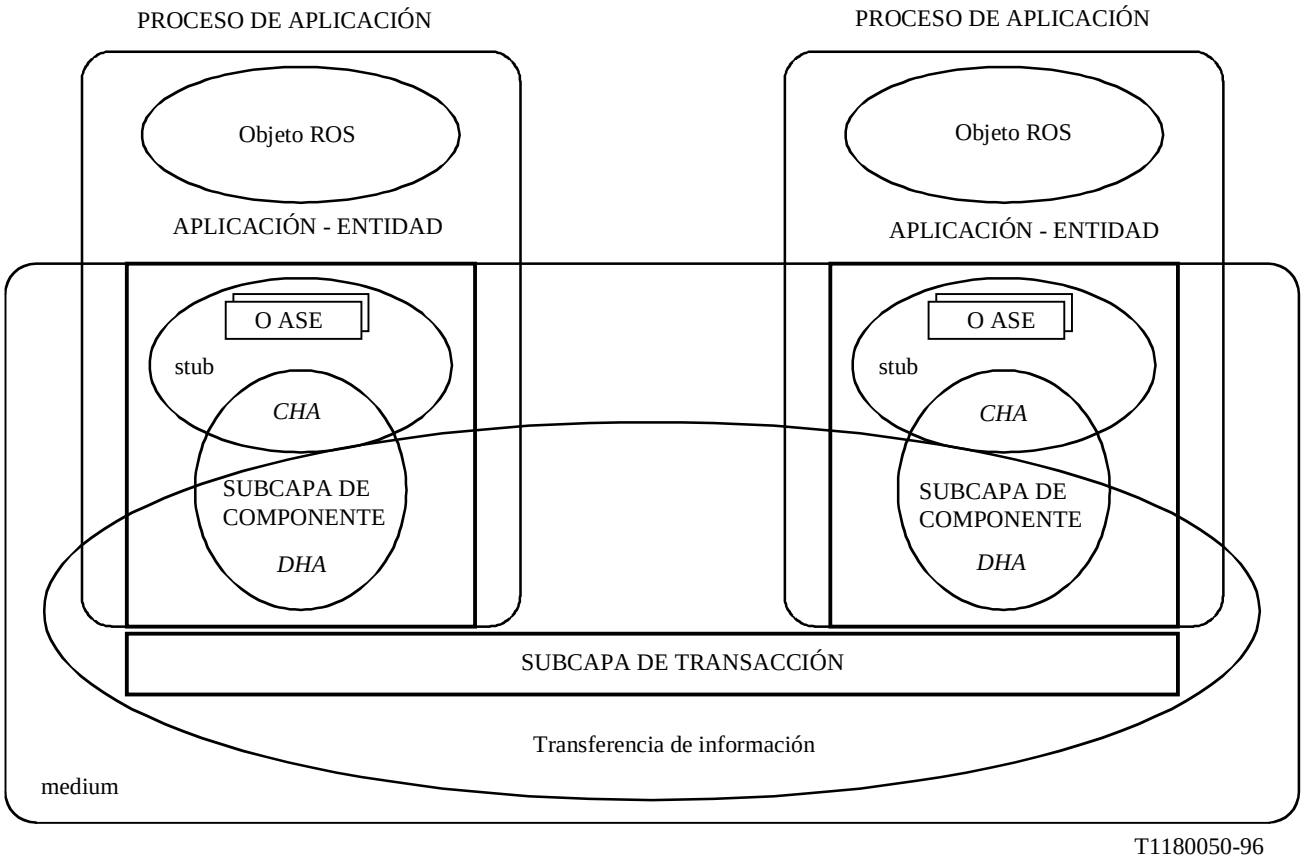


Figura 2/Q.775 – Realización TC de ROS

5.2 Realización del servicio de operaciones a distancia

5.2.1 Servicios básicos (stub)

La subcapa de componente TC proporciona los servicios necesarios para soportar la invocación de operaciones y para informar de las respuestas. También proporciona servicios locales adicionales para cancelar operaciones (petición TC-U-CANCELACIÓN, indicación TC-L-CANCELACIÓN) o para informar sobre errores de protocolo detectados localmente (indicación TC-L-RECHAZO).

Deben señalarse las restricciones siguientes:

- El conjunto de InvokeIds está restringido al rango de números enteros (–128 a 127).
- No se tiene en cuenta el campo &synchronous de la definición de operación. Desde un punto de vista de TC las operaciones siempre se consideran como asíncronas. Sin embargo, el usuario TC puede comportarse de forma síncrona.
- Los campos de prioridad de una definición de operación no se tienen en cuenta[1] .

5.2.2 Operaciones de vinculación y desvinculación

La TC no proporciona ningún mecanismo específico para invocar operaciones de vinculación y desvinculación. Es el usuario TC quien debe construir las APDU de vinculación y desvinculación y transferirlas a la TC como cualquier otro tipo de información de usuario. En consecuencia, la TC no es consciente de que se invoquen dichas operaciones y no verifica que se utilizan consistentemente en relación con el servicio de diálogo y el servicio de tratamiento de componentes (por ejemplo, no

[1] Esto puede evolucionar conforme progresen los estudios sobre tratamiento de prioridades en el SS N.º 7.

puede verificar que no se requiere ninguna operación después de que se haya invocado una operación de desvinculación).

5.2.2.1 Operación de vinculación

Cuando una definición de contexto de aplicación incluye un paquete de conexión, el usuario TC que inicia la operación invoca una operación de vinculación que se ejecuta como parte del procedimiento de establecimiento de diálogo previo a la ejecución de cualquier otra aplicación. El fracaso en la ejecución de esta operación provoca el rechazo del diálogo.

Si el usuario TC no necesita realmente invocar una operación de vinculación explícita, se supone que utiliza la operación predefinida emptyBind.

5.2.2.1.1 Invocación de una operación de vinculación

El usuario TC puede invocar una operación de vinculación utilizando la primitiva petición TC-COMIENZO. Si la definición de la operación de vinculación incluye un campo &ArgumentType, el usuario TC construye una PDU de vinculación-invocación a partir de dicha información y la transfiere como el primer (o parte del) parámetro de información de usuario de la primitiva petición TC-COMIENZO. En cualquier otro caso no se envía ninguna PDU de vinculación-invocación.

NOTA – Debe asegurarse que la PDU petición de vinculación se incluye en el primer campo externo del elemento de información de usuario de la APDU petición de diálogo (AARQ).

5.2.2.1.2 Respuesta a una operación de vinculación

El usuario TC informa del resultado de una operación de vinculación mediante la primera primitiva de tratamiento de diálogo que emite.

De la ejecución exitosa de la operación de vinculación se informa con la primitiva petición TC-CONTINUACIÓN, o con la primitiva petición TC-FINALIZACIÓN si no es necesario continuar con el diálogo. En este último caso también invocará una operación de desvinculación.

NOTA – El uso en esta etapa de la primitiva petición TC-FINALIZACIÓN limita la utilización de operaciones de desvinculación. Implica que solo el respondedor puede desvincular, que la definición de la operación de desvinculación no incluye un campo &ResultType y que la definición de su error asociado no incluye un campo &ParameterType (por ejemplo, como la operación emptyUnbind).

Si la definición de la operación de vinculación incluye un campo &ResultType, el usuario TC construye una PDU vinculación-resultado a partir de dicha información y la transfiere como la primera (o única) parte del parámetro de información de usuario de la primitiva petición TC-CONTINUACIÓN o bien de la primitiva petición TC-FINALIZACIÓN. En cualquier otro caso no se envía ninguna PDU vinculación-resultado.

El usuario TC informa de la ejecución infructuosa de una operación de vinculación utilizando la primitiva petición TC-U-ABORTO, que se emite como respuesta inmediata a la primitiva indicación TC-COMIENZO. El parámetro motivo de aborto toma el valor "diálogo rechazado" (dialogue-refused).

Si la definición del error asociado incluye un campo &ParameterType, el usuario TC construye una PDU vinculación-error a partir de dicha información y la transfiere como la primera (o única) parte del parámetro de información de usuario de la primitiva petición TC-CONTINUACIÓN o bien de la primitiva petición TC-FINALIZACIÓN. En cualquier otro caso no se envía ninguna PDU vinculación-error.

La operación emptyBind y las PDU vinculación-invocación, vinculación-resultado y vinculación-error se definen en la Recomendación X.880. Sus definiciones en ASN.1 se incluyen a continuación:

```
Bind {OPERATION:operation} ::= CHOICE
{
        bind-invoke             [16]   OPERATION.&ArgumentType (operation),
        bind-result             [17]   OPERATION.&ResultType (operation),
        bind-error              [18]   OPERATION.&Errors.&ParameterType (operation)
}

emptyBind OPERATION ::= {ERRORS {refuse} SYNCHRONOUS TRUE}
```

Donde *operación* se refiere a la operación de vinculación.

5.2.2.2 Operación de desvinculación

5.2.2.1.1 Invocación de una operación de desvinculación

Si la definición del contexto de aplicación incluye un paquete de conexión, el usuario TC invoca una operación de desvinculación como parte del procedimiento de terminación del diálogo.

La correspondencia en servicios TC depende del tipo de esta operación de desvinculación:

a) Si la definición de la operación de desvinculación no incluye un campo &ResultType y la definición de su error asociado no incluye un campo &ParameterType, la operación puede invocarse utilizando la primitiva petición TC-FINALIZACIÓN.

b) Si la definición de la operación de desvinculación incluye un campo &ResultType o la definición de su error asociado incluye un campo &ParameterType, la operación debe invocarse utilizando la primitiva petición TC-CONTINUACIÓN emitida por el peticionario de la desvinculación.

En ambos casos si la definición de la operación de desvinculación incluye un campo &ArgumentType, el usuario TC construye una APDU desvinculación-petición que se transfiere como la última (o la única) parte del parámetro de información de usuario de una primitiva petición TC-FINALIZACIÓN. En cualquier otro caso, no se envía ninguna APDU desvinculación-petición.

5.2.2.2.2 Respuesta a una operación de desvinculación

Cuando acepta una operación de desvinculación, el usuario TC emite una primitiva petición TC-FINALIZACIÓN. Si la definición de la operación de desvinculación incluye un campo &ResultType, el usuario TC construye una APDU desvinculación-resultado y la transfiere en la última (o única) parte del parámetro de información de usuario de la primitiva petición TC-FINALIZACIÓN. En cualquier otro caso no se envía ninguna APDU desvinculación-resultado.

Cuando rechaza una operación de desvinculación, el usuario TC emite una primitiva petición TC-CONTINUACIÓN. Si la definición del error asociado incluye un campo &ParameterType, el usuario TC construye una APDU vinculación-error y la transfiere como la última (o única) parte del parámetro de información de usuario de la primitiva petición TC-FINALIZACIÓN. En cualquier otro caso no se envía ninguna APDU vinculación-resultado.

NOTA – Esto debe garantizar que la PDU vinculación-resultado se incluye en el último campo externo del elemento de información de usuario de la APDU de respuesta de diálogo (AARE) cuando la primitiva petición TC-FINALIZACIÓN se emite como respuesta inmediata a la primitiva indicación TC-COMIENZO, o bien en el único campo EXTERNAL de DialoguePortion.

Si el contrato de asociación incluye un paquete de conexión pero el usuario TC no necesita invocar explícitamente una operación de desvinculación, se supone que se utiliza la operación emptyUnbind. Esta operación se corresponde conceptualmente en la primitiva petición TC-FINALIZACIÓN, aunque no se envía ninguna PDU de desvinculación.

La operación emptyUnbind y las PDU desvinculación-invocación, desvinculación-resultado y desvinculación-error se definen en la Recomendación X.880. Sus definiciones en ASN.1 se incluyen a continuación:

```
Unbind {OPERATION:operation} ::= CHOICE
{
        unbind-invoke       [19]   OPERATION.&ArgumentType (operation),
        unbind-result       [20]   OPERATION.&ResultType (operation),
        unbind-error        [21]   OPERATION.&Errors.&ParameterType (operation)
}

emptyUnbind OPERATION ::= {SYNCHRONOUS TRUE}
```

Donde *operación* se refiere a la operación de desvinculación.

5.3 Transferencia de información

5.3.1 Asociación de realizaciones

La TC proporciona dos realizaciones de asociación a través de su función de tratamiento de diálogo: el modo estructurado y el modo no estructurado que se definen en la Recomendación Q.771.

5.3.2 Realización de transferencia

En lo que a las operaciones a distancia se refiere, la TC proporciona al usuario las siguientes capacidades de transferencia de información:

- Las PDU de vinculación y desvinculación se transfieren en la información de usuario de la parte de diálogo (Dialogue Portion).
- Las PDU ROS básicas (más la devolución de resultado) se transfieren en la parte componente de cualquier mensaje.

La TC proporciona sólo un tipo de realización de transferencia, con independencia del tipo de realización de asociación elegida. Sin embargo, desde un punto de vista del emisor, esta realización ofrece cierta flexibilidad al usuario TC en lo que a la concatenación de las PDU se refiere.

Además de las operaciones a distancia, la TC ofrece los medios para transferir cualquier tipo de información de usuario mediante la utilización de primitivas de servicio de tratamiento de diálogo.

5.4 El contexto de aplicación basado en las TC

Los aspectos estáticos de la definición del contexto de aplicación basado en la TC que realiza algún contrato de asociación particular puede describirse como un objeto de información de clase APPLICATION-CONTEXT, que se especifica de la forma siguiente:

```
APPLICATION-CONTEXT ::=                          CLASS
{
        &associationContract                     CONTRACT,
        &dialogueMode                            DialogueMode,
        &termination                             Termination OPTIONAL,
        &componentGrouping                       BOOLEAN DEFAULT TRUE,
        &dialogueAndComponentGrouping            BOOLEAN DEFAULT TRUE,
        &AdditionalASEs                          OBJECT IDENTIFIER OPTIONAL,
        &AbstractSyntaxes                        ABSTRACT-SYNTAX,
        &applicationContextName                  OBJECT IDENTIFIER UNIQUE
}
```

```
WITH SYNTAX
{
        CONTRACT                                &associationContract
        DIALOGUE MODE                           &dialogueMode
        [TERMINATION                            &termination]
        [COMPONENT GROUPING ALLOWED             &componentGrouping]
        [DIALOGUE WITH COMPONENTS ALLOWED       &dialogueAndComponentGrouping]
        [ADDITIONAL ASEs                        &AdditionalASEs]
        ABSTRACT SYNTAXES                       &AbstractSyntaxes
        APPLICATION CONTEXT NAME                &applicationContextName
}
DialogueMode ::= ENUMERATED {structured (1), unstructured (2)}

Termination ::= ENUMERATED {basic (1), prearranged (2)}
```

El campo &associationContract identifica el contrato de asociación que realiza este contexto de aplicación.

El campo &dialogueMode indica si este contexto de aplicación utiliza las facilidades de diálogo en modo estructurado o las facilidades de diálogo en modo no estructurado. Si la definición del contrato de asociación incluye un paquete de conexión, el campo &dialogueMode indicará "estructurado" (structured).

El campo &termination indica si para finalizar el diálogo se utiliza una terminación básica o una previamente dispuesta. Si este campo no está presente, la definición del contexto de aplicación no pone limitación alguna sobre cuál es el método de finalización que se utiliza.

El campo &componentGrouping indica si varios componentes pueden agruparse en un único mensaje. Si este campo no está presente, la definición del contexto de aplicación no pone limitación alguna sobre este asunto.

El campo &dialogueAndComponentGrouping indica si las PDU de vinculación y de desvinculación pueden enviarse en mensajes que también contienen componentes. Si este campo no está presente, la definición del contexto de aplicación no pone limitación alguna sobre este asunto.

El campo &AdditionalASEs contiene los identificadores de objeto de los ASE requeridos por el contexto de aplicación (si es que hay alguno) que no se basan en la utilización de operaciones a distancia.

El campo &AbstractSyntaxes contiene las sintaxis abstractas que se requieren para transportar la información entre objetos, incluyendo las PDU para la invocación e información de las operaciones del contrato.

El campo &applicationContextName contiene el valor que debe proporcionarse a la TC para identificar el contexto de aplicación.

5.5 Sintaxis abstractas

5.5.1 Control del diálogo

El campo &AbstractSyntaxes de una definición de contexto de aplicación debe incluir la siguiente sintaxis abstracta si el campo &dialogueMode indica "estructurado".

```
dialogue-abstract-syntax ABSTRACT-SYNTAX ::= {DialoguePDU IDENTIFIED BY dialogue-as-id}
```

El campo &AbstractSyntaxes de una definición de contexto de aplicación debe incluir la siguiente sintaxis abstracta si el campo &dialogueMode indica "no estructurado" (unstructured).

```
uniDialogue-abstract-syntax ABSTRACT-SYNTAX ::= {UniDialoguePDU IDENTIFIED BY uniDialogue-as-id}
```

5.5.2 Sintaxis definida por el usuario

5.5.2.1 Generalidades

El campo &AbstractSyntaxes de una definición de contexto de aplicación debe incluir una o más sintaxis abstractas para representar los mensajes TC (incluyendo los componentes) y las PDU de vinculación y de desvinculación. Dichas sintaxis abstractas deben ser definidas por el diseñador de la aplicación.

Los mensajes TC se definen en la Recomendación Q.773 mientras que las PDU de vinculación y de desvinculación se definen en la Recomendación X.880.

Se deja al criterio del diseñador determinar el número de sintaxis abstractas que se definen para soportar un contexto de aplicación particular. Sin embargo, deben seguirse las reglas siguientes:

a) Si el contexto de aplicación realiza un contrato de asociación que incluye un paquete de conexión, los valores de los tipos de datos:

Bind{ac.&associationContract.&connection.&bind}
Unbind{ac.&associationContract.&connection.&unbind}

aparecerán en al menos una de dichas sintaxis abstractas.

b) Para cada operación *op* involucrada en el conjunto del paquete de operaciones utilizado por el contexto de la aplicación, habrá al menos una sintaxis abstracta que incluirá los valores de los tipos siguientes:

Invoke {TCInvokeIds, OPERATION:op}
ReturnResult {OPERATION:op}

c) Para cada error *err* involucrado en el conjunto del paquete de operaciones utilizado por el contexto de la aplicación, habrá al menos una sintaxis abstracta que incluirá los valores de los tipos siguientes:

ReturnError {ERROR:err}

d) Al menos una sintaxis abstracta incluirá :

Reject

5.5.2.2 Definición de las sintaxis abstractas

Dado un paquete de operación puede definirse una sintaxis abstracta que permita el intercambio de mensajes TC que transporten invocación e información de todas sus operaciones utilizando los siguientes tipos de datos:

TCSingleAS {OPERATION-PACKAGE: package} ::= TCMessage { {AllOperations {package}}, {AllOperations {package}} }

Alternativamente, pueden definirse un par de sintaxis abstractas en base a los dos tipos siguientes:

TCConsumerAS {OPERATION-PACKAGE: package} ::= TCMessage { {ConsumerPerforms {package}}, {ConsumerPerforms {package}} }

TCSupplierAS {OPERATION-PACKAGE: package} ::= TCMessage { {SupplierPerforms {package}}, {SupplierPerforms {package}} }

Una única sintaxis abstracta puede acomodar un conjunto de paquetes siempre que los códigos de error y de operación sean únicos. Por ejemplo, el siguiente tipo de datos puede utilizarse como base de una sintaxis abstracta para acomodar todos los paquetes de operación involucrados en un contrato de asociación:

```
AllPackagesAS {APPLICATION-CONTEXT:ac} ::=
        TCSingleAS
        {
            combine
                {
                    {
                        ac.&associationContract.&OperationsOf
                        | ac.&associationContract.&InitiatorConsumerOf
                        | ac.&associationContract.&InitiatorSupplierOf
                    }
                    {},
                    {}
                }
        }
```

En base a los tipos siguientes puede definirse una sintaxis abstracta para representar los valores de PDU de vinculación y desvinculación:

```
ConnectionAS {APPLICATION-CONTEXT:ac} ::= CHOICE
{
    bind        Bind{ac.&associationContract.&connection.&bind},
    unbind      Unbind{ac.&associationContract.&connection.&unbind}
}
```

El valor del identificador de objeto asignado a esta sintaxis abstracta debe ser incluido en el campo de múltiples valores &abstract-syntax-name de la definición del contexto de aplicación. Ello pretende servir de referencia directa cuando los valores de dichas PDU se transporten en el parámetro de información de usuario si se trata de PDU de control de diálogo (Dialogue Control PDUs) o directamente como valor de la parte de diálogo.

También pueden definirse sintaxis abstractas adicionales para representar valores de las PDU asociadas con los ASE no basados en ROS (véase un ejemplo en 4.7).

5.6 Ampliación de la notación

El siguiente módulo ASN.1 contiene definiciones que permiten al diseñador de un protocolo usuario-TC especificar contextos de aplicación y sintaxis abstractas como instancias de clases de objetos de información adecuadas.

```
TC-Notation-Extensions  {ccitt recommendation q 775 modules(2) notation-extension(4) version1(1)}

DEFINITIONS ::=

BEGIN

IMPORTS

TCMessage{} FROM TCAPMessages {ccitt recommendation q 773 modules(2) messages(1) version3(3)}

Bind{}, Unbind{} FROM Remote-Operations-Generic-ROS-PDUs {joint-iso-ccitt remote-operations(4) generic-
ROS-PDUs(6) version1(0)}

AllOperations{}, ConsumerPerforms{}, SupplierPerforms{}, combine{} FROM Remote-Operations-Useful-
Definitions {joint-iso-ccitt remote-operations(4) useful-definitions(7) version1(0)}

CONTRACT, OPERATION-PACKAGE FROM Remote-Operations-Information-Objects {joint-iso-ccitt remote-
operations(4) informationObjects(5) version1(0)}
```

```
UniDialoguePDU, uniDialogue-as-id FROM UnidialoguePDUs
        {ccitt recommendation q 773 modules(2) unidialoguePDUs(3) version1(1)}

DialoguePDU, dialogue-as-id FROM DialoguePDUs
        {ccitt recommendation q 773 modules(2) dialoguePDUs(2) version1(1)}

APPLICATION-CONTEXT ::=                                 CLASS
{
        &associationContract                            CONTRACT,
        &dialogueMode                                   DialogueMode,
        &termination                                    Termination OPTIONAL,
        &componentGrouping                              BOOLEAN DEFAULT TRUE,
        &dialogueAndComponentGrouping                   BOOLEAN DEFAULT TRUE,
        &AdditionalASEs                                 OBJECT IDENTIFIER OPTIONAL,
        &AbstractSyntaxes                               ABSTRACT-SYNTAX,
        &applicationContextName                         OBJECT IDENTIFIER UNIQUE
}

WITH SYNTAX
{
        CONTRACT                                        &associationContract
        DIALOGUE MODE                                   &dialogueMode
        [TERMINATION                                    &termination]
        [COMPONENT GROUPING ALLOWED                     &componentGrouping]
        [DIALOGUE WITH COMPONENTS ALLOWED               &dialogueAndComponentGrouping]
        [ADDITIONAL ASEs                                &AdditionalASEs]
        ABSTRACT SYNTAXES                               &AbstractSyntaxes
        APPLICATION CONTEXT NAME                        &applicationContextName
}

DialogueMode ::= ENUMERATED {structured (1), unstructured (2)}

Termination ::= ENUMERATED {basic (1), prearranged (2)}

dialogue-abstract-syntax ABSTRACT-SYNTAX ::= {DialoguePDU IDENTIFIED BY dialogue-as-id}

uniDialogue-abstract-syntax ABSTRACT-SYNTAX ::= {UniDialoguePDU IDENTIFIED BY uniDialogue-as-id

TCSingleAS {OPERATION-PACKAGE: package} ::= TCMessage { {AllOperations {package}}, {AllOperations
{package}} }

TCConsumerAS {OPERATION-PACKAGE: package} ::= TCMessage { {ConsumerPerforms {package}},
{ConsumerPerforms {package}} }

TCSupplierAS {OPERATION-PACKAGE: package} ::= TCMessage { {SupplierPerforms {package}},
{SupplierPerforms {package}} }

AllPackagesAS {APPLICATION-CONTEXT:ac} ::=
        TCSingleAS
                {
                        combine
                                {
                                        {
                                                ac.&associationContract.&OperationsOf
                                                | ac.&associationContract.&InitiatorConsumerOf
                                                | ac.&associationContract.&InitiatorSupplierOf
                                        },
                                        {},
                                        {}
                                }
```

```
        }
ConnectionAS {APPLICATION-CONTEXT:ac} ::= CHOICE
{
    bind        Bind{ac.&associationContract.&connection.&bind},
    unbind      Unbind{ac.&associationContract.&connection.&unbind}
}
END
```

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	Transmisiones de señales radiofónicas, de televisión 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	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
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 y comunicación entre sistemas abiertos
Serie Z	Lenguajes de programación

Ginebra, 1997