UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

Serie Q

Suplemento 29
(12/1999)

SERIE Q: CONMUTACIÓN Y SEÑALIZACIÓN

Modelado de servicios: Evolución al uso de técnicas orientadas a objetos

Recomendaciones UIT-T de la serie Q – Suplemento 29

(Anteriormente Recomendaciones del CCITT)

RECOMENDACIONES UIT-T DE LA SERIE Q

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
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.1699
REQUISITOS Y PROTOCOLOS DE SEÑALIZACIÓN PARA IMT-2000	Q.1700–Q.1799
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.

Suplemento 29 a las Recomendaciones UIT-T de la serie Q

Modelado de servicios: Evolución al uso de técnicas orientadas a objetos

Resumen

Este Suplemento compara diferentes tipos de metodologías para el modelado de servicios con el fin de determinar su idoneidad en relación con la elaboración de protocolos para la red inteligente en el marco del conjunto de capacidades 4 de red inteligente. Investiga también la evolución de las técnicas basadas en bloques de construcción independientes del servicio y considera diferentes tecnologías, tales como las interfaces de programación de aplicación.

Este Suplemento complementa la información contenida en la Recomendación Q.65.

Finalidad

La finalidad de este Suplemento es propiciar el análisis de los aspectos de modelado de servicios relacionados con el conjunto de capacidades 4 de la red inteligente.

Orígenes

El Suplemento 29 a las Recomendaciones UIT-T de la serie Q, preparado por la Comisión de Estudio 11 (1997-2000) del UIT-T, fue aprobado por el procedimiento de la Resolución 5 de la CMNT el 3 de diciembre de 1999.

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 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 publicación, 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 publicación 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 publicaciones.

En la fecha de aprobación de la presente publicación, la UIT no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta publicación. 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

Suplemento 29 a las Recomendaciones UIT-T de la serie Q

Modelado de servicios: Evolución al uso de técnicas orientadas a objetos

1 Alcance del modelado de servicios para el conjunto de capacidades 4 de RI

La finalidad de este Suplemento es:

- identificar y comparar diferentes metodologías para el modelado de servicios;
- determinar la idoneidad de las metodologías identificadas para el modelado de servicios, en particular con respecto a:
  - gestión de servicio: modelado de datos
  - fines de la lógica de servicio: modelo de servicio dinámico (comportamiento de la lógica de servicio)
  - descripción de la relación entre el modelo de datos y el modelo de servicio dinámico.
- determinar la idoneidad de las metodologías identificadas para el desarrollo de protocolos, en particular, con respecto a:
  - el soporte del refinamiento gradual de las capacidades de servicios y de redes a nivel de protocolo, para el modelo de datos y el modelo de servicio dinámico
  - la posibilidad de incorporar el protocolo existente en el modelo definido con la metodología utilizada
- investigar los aspectos de migración del modelado de servicios, en particular, con respecto a:
  - la evolución del modelado de servicios basado en bloques de construcción independientes del servicio (SIB) a las metodologías utilizadas para el conjunto de capacidades 4 de red inteligente.
- decidir el alcance del modelado de servicios para el conjunto de capacidades 4 de red inteligente
- decidir la metodología que se ha de aplicar para el modelado de servicios en el conjunto de capacidades 4 de red inteligente.

1.1 Referencias

Los siguientes Informes técnicos y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones del presente Suplemento. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todos los Suplementos y otras referencias son objeto de revisiones, con lo que se preconiza que los participantes en acuerdos basados en el presente Suplemento investiguen la posibilidad de aplicar las ediciones más recientes de los Suplementos y otras referencias citadas a continuación. Se publica regularmente una lista de las Recomendaciones UIT-T actualmente vigentes.

[1] Recomendaciones UIT-T de la serie Q.122x, *Recomendaciones relativas al conjunto de capacidades 2 de red inteligente*.

[2] Recomendación UIT-T Q.1223 (1997), *Plano funcional global para el conjunto de capacidades 2 de red inteligente*.

[3] Recomendación UIT-T Z.100 (1999), *Lenguaje de especificación y descripción*.

2 Definiciones y abreviaturas

2.1 Abreviaturas

En este Suplemento se utilizan las siguientes siglas.

API Interfaz programable de aplicación (*application programming interface*)

GFP Plano funcional global (*global functional plane*)

INCM Modelo conceptual de red inteligente (*intelligent network conceptual model*)

ODP Procesamiento distribuido abierto (*open distributed processing*)

OMT Técnica de modelado de objetos (*object modelling technique*)

SDL Lenguaje de especificación y descripción (*specification description language*)

SQL Lenguaje de indagación estructurado (*structured query language*)

STD Diagramas de transición de estados (*state transition diagrams*)

TINA Arquitectura de interfuncinamiento de redes de información de telecomunicaciones (*telecommunication information networking architecture*)

2.2 Definiciones

En este Suplemento se definen los términos siguientes:

2.2.1 **API**: Es esencialmente un conjunto de operaciones (o *métodos*) que pueden ser invocados en un componente, cada una de las cuales hace que el componente presente una funcionalidad comportamental. Cada operación es especificada sintácticamente como un identificador que identifica la operación que se invoca y parámetros que afectan el comportamiento del componente de alguna manera.

2.2.2 **llamada API**: Equivale al término "operación" y algunas veces se utiliza de manera intercambiable en la descripción de una API (véase más arriba). Una API está formada efectivamente por varias llamadas API.

3 Requisitos para el modelado de servicios del CS-4 de RI

Esta cláusula trata de los requisitos que se consideran importantes para las técnicas y metodologías de modelado utilizadas para el modelado de servicios de red inteligente (RI), a saber:

- Sustentación del **refinamiento gradual** del servicio (características) al nivel de protocolo.
- Sustentación de varios mecanismos de **transparencia**, entre los que cabe citar:
  - Transparencia de *tecnología*: El modelado de los servicios de RI no debe depender de la tecnología utilizada, por ejemplo, el tipo de red, el sistema operativo o el lenguaje de programación. Un tipo de transparencia importante en el contexto del modelado de servicios es la transparencia de tecnología de red. El objetivo del CS-4 de RI es la integración de varias tecnologías de red. incluido los tipos de redes con conexión (por ejemplo, RTPC, IMT-2000) y sin conexión (por ejemplo, redes basadas en IP, redes de datos). Probablemente algunos de los servicios previstos en el CS-4 de RI serán específicos de una tecnología de red, otros requerirán el interfuncionamiento de diferentes tecnologías de red, o su comportamiento dependerá de la tecnología del servicio invocado. Cabría imaginar un servicio con interacción de usuario que pudiera ser invocado por la RDSI de banda ancha y por usuarios de redes móviles, o un servicio de conferencia multimedios por diferentes tecnologías de red. La RI podrá ser una arquitectura integrada para los servicios transmitidos por múltiples tecnologías de red.

Se debe investigar en qué medida es posible lograr la transparencia de la tecnología de red.

- Transparencia de *acceso*: Enmascaramiento de diferencias en mecanismos de representación e invocación de datos (que proporcionan, por ejemplo, múltiples correspondencias del contenido de información de intercambio de protocolo de RI a API).
- Transparencia de *fallos*: Enmascaramiento, a partir de un objeto, del fallo y posible restablecimiento de objetos (incluido el propio objeto).
- Transparencia de *migración*: Enmascaramiento, a partir de un objeto, de la capacidad de un sistema para cambiar la ubicación de las interfaces para ese objeto.

  Transparencia de *reubicación*: Enmascaramiento de la reubicación de una interfaz de objeto a partir de otras interfaces vinculadas a éste.
- Transparencia de *replicación*: Enmascaramiento del uso de un grupo de objetos compatibles mutuamente comportamentales para sustentar una interfaz.

- **Extensibilidad de servicios**: Debe ser posible sustentar adiciones a la funcionalidad de servicio de la RI extendiendo los modelos de servicio existentes y los componentes de servicio existentes.
- **Mejora gradual de servicios**: Debe ser posible acomodar y aumentar las capacidades de servicio desde el punto de vista del número de usuario, número de nodos, número de dominios administrativos, etc.
- **Tratamiento de versiones de servicio:** Las técnicas de modelado de servicios deben facilitar el uso concurrente de múltiples versiones de modelos y componentes de servicio.
- **Reutilización de servicios**: Debe ser posible reutilizar los modelos de un servicio (capacidad) RI durante la especificación de otro servicio RI, en vez de comenzar el modelo desde el principio, incluso en caso de funcionalidad de superposición. Lo mismo se aplica a los componentes de soporte lógico que aplican la especificación.
- **Operación, administración y mantenimiento**: Las metodologías y técnicas de modelado deben sustentar la flexibilidad cuando la funcionalidad y el modelo de datos cambian durante la operación, administración y mantenimiento. Por ejemplo, los cambios de funcionalidad que se producen durante el mantenimiento deben ser reflejados fácilmente en el modelo del servicio de RI.
- **Reducción de conflicto de servicios**: Las metodologías y las técnicas de modelado deben reducir los riesgos de interacciones de servicios.
- **Modificación/adaptación flexible de servicios**: Los proveedores de redes y de servicio deben ser capaces de diferenciar y adaptar (personalizar) los servicios para satisfacer necesidades específicas del mercado.

  Los usuarios deben ser capaces de adaptar los servicios en cierta medida para satisfacer sus necesidades personales. A corto plazo esto estará limitado a la personalización de los datos y a la selección de una lista de prestaciones predefinidas, pero a más largo plazo, se podrá proporcionar también "paquetes" de servicios.
- **Soporte de un entorno de múltiples partes interesadas**. El modelado de capacidades de servicios del CS-4 de RI debe tener en cuenta los respectivos cometidos de las diferentes partes interesadas (stakeholders) que participan en el suministro de servicios (detallista, entidad operadora, proveedor de contenido, etc.).

3.1 Modelado de servicios

Uno de los objetivos del modelado orientado a objetos (OO) es permitir que el proyectista y los diseñadores del servicio compartan el mismo modelo de lógica de servicio. Para lograr esto, la técnica de modelado debe ocultar al usuario todos los detalles de la plataforma y permitir al mismo tiempo al proyectista del servicio que traduzca el modelo de comportamiento en la forma de ejecución adecuada. Con el fin de satisfacer estos requisitos, se propone que la técnica de modelado se represente en una notación que sea independiente del lenguaje de programación, del sistema operativo y de la base de datos. Se propone el uso de diagramas SDL y de transición de estados sin datos, para representar el comportamiento puro de una clase abstracta.

3.2 Lógica de servicio que abarca una sola clase

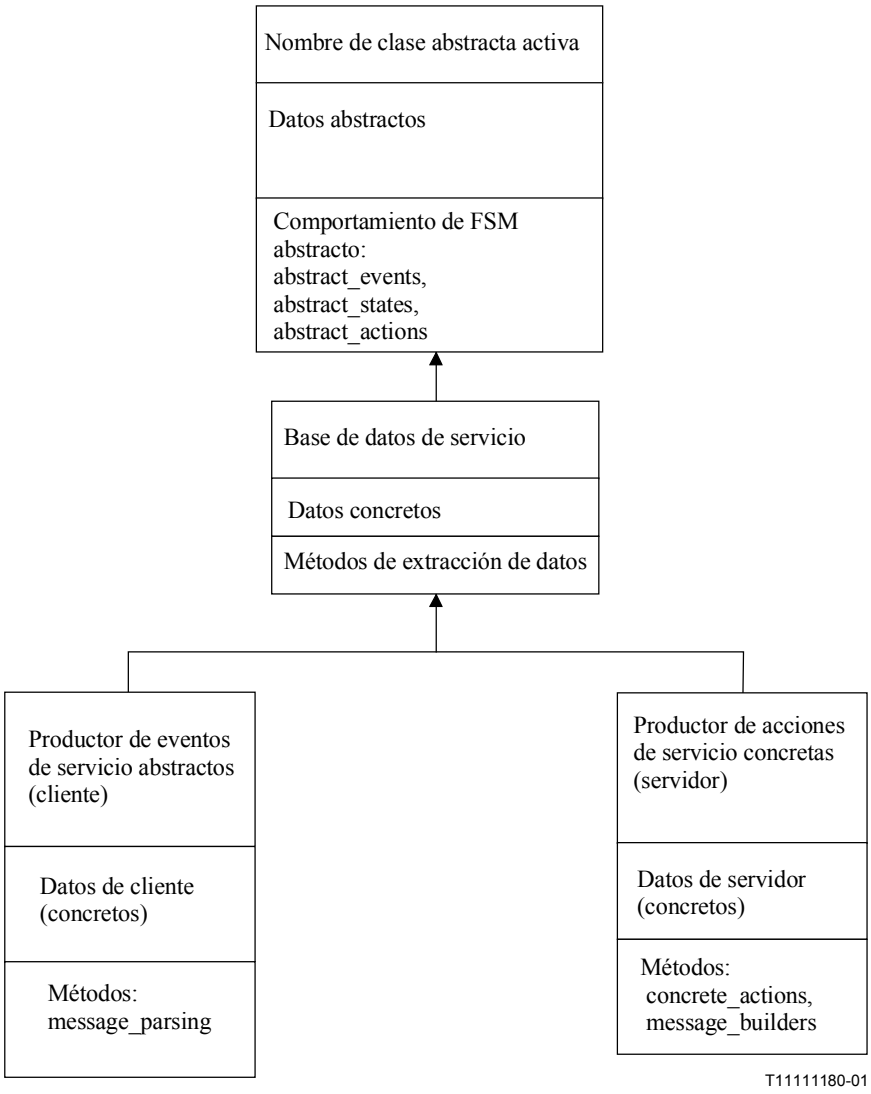
OMT y otras técnicas de modelado introducen el concepto de una clase activa, que se define como una clase cuyo comportamiento se modela mejor usando el modelo de máquina de estados finitos (*FSM, finite state machine model*). Para lograr la independencia del lenguaje a nivel de lógica de servicio se propone que el modelo de máquina de estados finitos no contenga datos. De este modo, una clase abstracta que representa un simple servicio contendría un modelo de datos abstractos y de comportamiento abstracto estrictamente separados.

Es importante destacar que la separación de un modelo de datos abstractos y de comportamiento abstracto se sugiere sólo a nivel del modelado. No se hacen recomendaciones ni se sugiere que la separación de un modelo de datos abstractos y de comportamiento abstracto se extienda al código generado automática o manualmente.

Un ejemplo de separación de un modelo de datos abstractos y de comportamiento abstracto podría ser un servicio de red privada virtual (RPV) con un modelo de servicio complejo. Por ejemplo, el parámetro de servicio de red privada virtual Participant_ID puede ser un ejemplo de datos abstractos a nivel de modelado. El modelo de clase abstracta en su sección de datos abstractos no debe reflejar dónde se utilizan estos datos en el modelo comportamental. Por otra parte, el modelo de comportamiento abstracto de una clase abstracta puede requerir un New_Participant_Arrived sin tener que mencionar dónde se almacenan los datos para un participante dado en la sección de datos abstractos de la clase abstracta.

Esta separación dentro de una clase de alto nivel debe permitir la intercambiabilidad de objetos del modelo de servicio y la creación de necesidades de servicio normalizadas. Véase la figura 1. Las flechas del modelo representan herencia de una manera coherente con la notación OMG. En el caso de aplicación de C++, abstract_actions y abstract_events se usan a menudo como funciones virtuales puras.

La figura 2 muestra un ejemplo de una clase activa abstracta denominada objeto de servicio de red privada virtual, que se puede utilizar para la realización del servicio de red privada virtual. La clase contiene ejemplos de abstract_events, abstract_actions y abstract_states.



NOTA – Es factible que el message_parser y el message_builder puedan ser realizados como clases separadas para servicios más complejos.

Figura 1 – Una clase abstracta que representa: a) Una lógica de servicio simple; b) Una porción de una lógica de servicio compleja; c) Un gestor de lógica de servicio

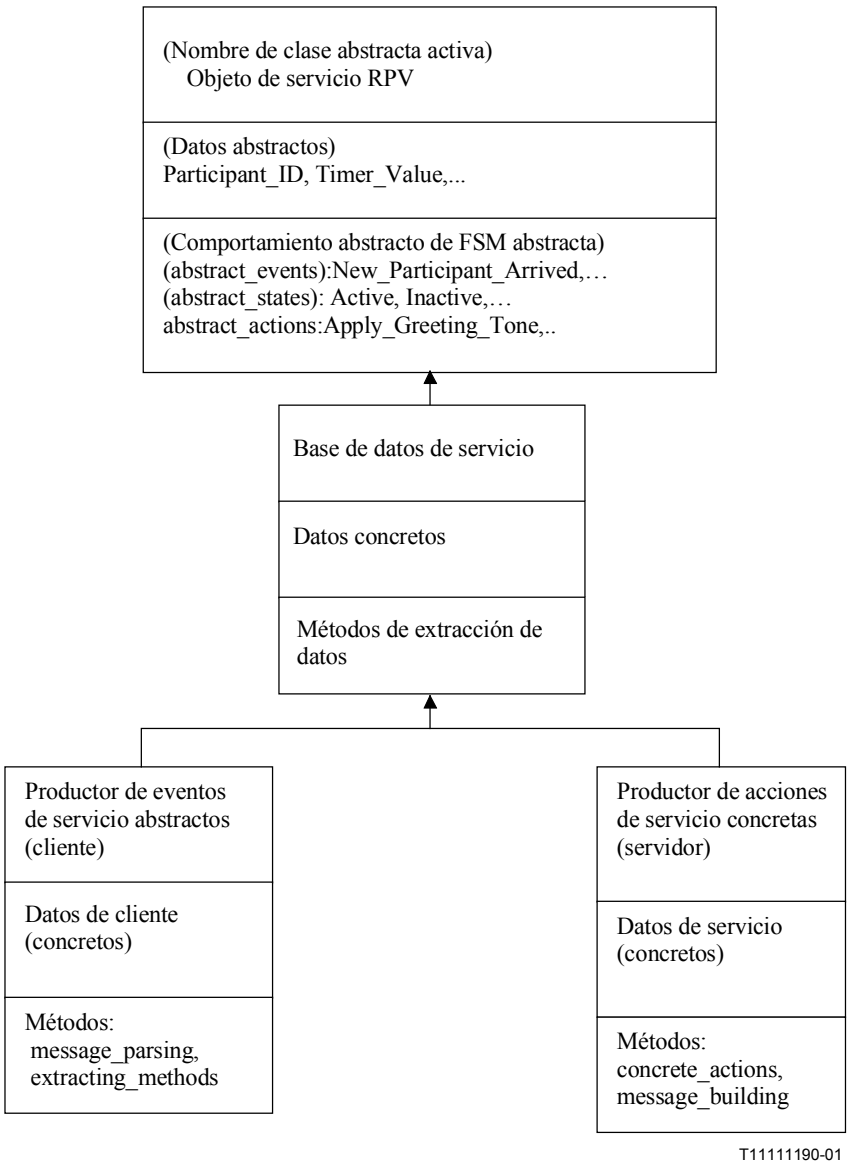


Figura 2 – Ejemplo de clase abstracta que representa una lógica de servicio de red privada virtual

3.2.1 Lógica de servicio que abarca varias clases

Para servicios más complejos, es posible modelar la lógica de servicio como un conjunto de clases activas que cooperan. Un objeto se define aquí como una clase ejemplificada. Puede ser útil investigar si el modelo de clases cooperantes es más apropiado para el modelado de servicio RI que el modelo de clases contenidas.

Hay indicaciones de que el modelo de clases cooperantes refuerza mejor el encapsulado de clases que oculta los detalles de realización. Este modelo de clases cooperantes fue una base del lenguaje de programación Smalltalk-Object. En este caso, cada una de las clases activas debe ser modelada de la misma manera que el caso de lógica de servicio que abarca una sola clase. Las clases cooperantes envían peticiones entre sí y reciben respuestas. Esto representa un flujo de información de control entre los objetos.

Cabe señalar que el término mensaje según se usa en este documento supone un flujo de información físico y no lógico. Un mensaje puede contener datos e información de control. Por consiguiente, se recomienda que el flujo de información de control entre los objetos se muestre separadamente del

flujo de datos entre los objetos. Una ventaja conocida de esta separación es una simplificación de las herramientas que validan el servicio automáticamente. La herramienta de validación del entorno de creación de servicios (SCE, *service creation environment*) debe generar diagramas de flujos de secuencias de comunicación entre objetos a partir de modelos de comportamiento FSM de objetos cooperantes. En fecha ulterior se podrán proporcionar otras contribuciones sobre los requisitos del SCE y SMS. El modelo presentado se aplicará a comunicaciones entre objetos y a la situación cuando se asigna a una de las clases una función del gestor de lógica de servicio. Esto corresponde con la elección del diseñador de servicios de utilizar una forma jerárquica o más bien descentralizada del modelado de servicios.

4 Metodologías y técnicas de modelado

4.1 Procesamiento distribuido abierto (ODP)

El modelo de referencia de ODP es una metodología distribuida genérica orientada a objetos que es adecuada para aplicaciones de telecomunicaciones tradicionales (como la RI) y para aplicaciones de procesamiento de información. Se basa en dos poderosas tendencias de la tecnología de soporte lógico:

- Técnicas de especificación orientadas a objetos proporcionadas en diferentes niveles de abstracción que permiten un alto grado de refinamiento gradual y comprobación de coherencia
- Entornos de procesamiento distribuido basados en objetos (por ejemplo, plataformas basadas en CORBA) que permiten la provisión de servicios distribuidos y, lo que es más importante, posibilitan la transparencia de distribución y el interfuncionamiento dentro de los sistemas distribuidos.

El modelo de referencia de ODP prescribe que se proporcionen múltiples descripciones, con diferentes niveles de abstracción, para cualquier servicio en estudio o en desarrollo. En el ODP se han definido cinco niveles de abstracción, denominados *puntos de vista* en la terminología ODP, y se considera que abarcan los diferentes sectores de interés que han de ser tratados en el proceso de desarrollo de servicios.

Hay que proporcionar especificaciones en cada uno de estos puntos de vistas, utilizando el correspondiente modelo de puntos de vista o un lenguaje de puntos de vista adecuado. Los cinco puntos de vista ODP son *empresa, información*, *computacional*, *ingeniería* y *tecnología*.

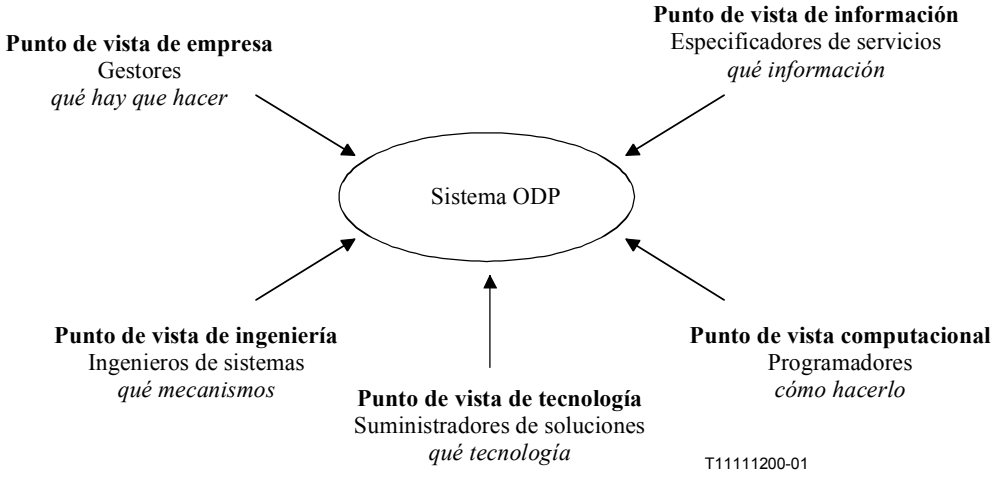


Figura 3 – Diferentes niveles de abstracción (puntos de vista) en ODP

4.1.1 Punto de vista de empresa

Una especificación de empresa de un servicio RI describe el servicio desde la perspectiva de la organización y de las personas que utilizarán o harán funcionar ese servicio. Los conceptos empleados para las especificaciones de empresa son los siguientes:

- son las necesidades las que indican las capacidades deseadas del servicio en un nivel estratégico (por qué se considera);
- las partes interesadas (y otras instituciones participantes);
- los cometidos (quién participará y con qué cometido);
- el entorno jurídico (obligaciones de las partes interesadas);
- el conjunto de reglas y reglamentaciones que rigen la explotación del servicio (las reglas que deben ser obedecidas cuando el servicio funcione).

Una especificación de empresa se suele escribir en formato textual o utilizando una notación gráfica de alto nivel.

4.1.2 Punto de vista de información

Una especificación de información proporciona un modelo muy abstracto de entidades del mundo real y sus relaciones junto con las restricciones que rigen su comportamiento. Una especificación de información abarca típicamente:

- la especificación de la estructura de información del servicio (objetos),
- a especificación de las dependencias entre objetos de información (asociaciones),
- la especificación de las operaciones y constricciones que rigen la evolución dinámica del servicio consideradas como un conjunto de objetos.

El punto de vista de información podrá ser desarrollado utilizando, por ejemplo, el lenguaje de modelado unificado (UML), especialmente las capacidades para el modelado de datos (modelo estático).

Orientaciones para la correspondencia entre el punto de vista de empresa y el punto de vista de información

- Es posible establecer reglas de correspondencia para traducir la especificación de empresa en términos de información. Se señala que sólo se trata de orientaciones.

4.1.3 Punto de vista computacional

Una especificación computacional representa una realización abstracta del servicio considerado desde el punto de vista de los objetos que interactúan, en la cual la ubicación, el acceso, la distribución y los fallos son transparentes. Dicho simplemente, en este nivel de abstracción un servicio se representa como una configuración dinámica de objetos de interacción.

Una característica importante de la especificación computacional es la capacidad de capturar los aspectos en tiempo real y probabilísticos. Por ejemplo, la vinculación de objetos describe el comportamiento de comunicación, que cumple ciertas constricciones de calidad de servicio. Estos requisitos no funcionales son particularmente pertinentes cuando se trata del soporte de interacciones multimedios en un entorno distribuido.

La especificación del punto de vista computacional contendría típicamente entidades funcionales bien conocidas, tales como función de control de servicio (SCF, *service control function*), función de conmutación de servicio (SSF, *service switching function*), función de recursos especializados (SRF, *specialized resource function*), etc., como objetos computacionales, quizá más descompuestos, dependiendo de las necesidades de distribución. Las interacciones entre los objetos computacionales serían "operaciones" [es decir, operaciones de protocolo de aplicación de red inteligente (INAP, *IN application protocol*)] y "flujos" (es decir, voz, vídeo y datos).

El punto de vista computacional se podría desarrollar utilizando, por ejemplo, el UML, para la especificación de los objetos computacionales y, por ejemplo, el lenguaje de definición de interfaces (IDL, *interface definition language*) para la especificación de interfaces entre objetos.

Aunque el ODP no recomienda ninguna solución de correspondencia entre tipos de objetos de información y computacionales, la siguiente orientación simple es un punto de partida: una correspondencia de uno a uno. Se obtiene una especificación de tipos de objetos computacionales tomando una especificación de tipos de objetos computacionales y añadiendo la descripción de las operaciones para las cuales puede tener un cometido de servidor. Sin embargo, las consideraciones relativas al potencial para la distribución modificarán esta correspondencia.

4.1.4 Punto de vista de ingeniería

Una especificación de ingeniería determina cómo se pueden realizar descripciones computacionales sin distribución en términos de componentes de sistema genéricos y protocolo de comunicaciones (como el sistema de señalización N.° 7). Por consiguiente, se centra en cómo lograr la interacción entre objetos y qué recursos se necesitan para ello. Desde el punto de vista de ingeniería, un sistema ODP se considera como un conjunto de sistemas de computador. Los detalles de las redes de comunicación subyacentes, sistemas operativos y soporte físico quedan ocultos por un entorno distribuido básico y uniforme.

En la terminología ODP, las SSF-FSM y los BCSM pueden ser considerados como *stubs* u *objetos de protocolo*. Las firmas de las operaciones corresponderían con las definiciones ASN.1 de las operaciones del protocolo de aplicación de red inteligente (INAP, *IN application protocol*).

La especificación de ingeniería sería muy similar a la especificación de protocolo.

Se podría elaborar utilizando el SDL y la ASN.1. El SDL se usa para describir los objetos y su comportamiento, y la ASN.1 para describir las firmas de las operaciones visibles en las interfaces externas (que corresponden con mensajes de protocolo). El motivo de utilizar la ASN.1 y no el IDL es que un gran número de protocolos que la RI necesita para interfuncionar están ya especificados con la ASN.1.

Se puede mantener la coherencia entre las especificaciones computacional y de ingeniería utilizando el soporte de herramientas. Por ejemplo, los objetos computacionales, especificados con UML/OMT, pueden ser "pegados" en la especificación SDL como objetos (tipos) SDL. La herramienta mantendrá estos enlaces entre objetos UML/OMT y SDL, y es posible comprobar la coherencia entre los dos modelos. Asimismo, es posible general automáticamente diagramas de procesos SDL a partir de descripciones de gráficos de estados UML/OMT.

4.1.5 Punto de vista de tecnología

El punto de vista tecnología describe la realización del sistema en términos de componentes de soporte físico y soporte lógico. Puede ser necesario considerar las constricciones de costo y disponibilidad. Las selecciones influyen en el funcionamiento y la calidad de servicio del sistema. Como está directamente relacionado con la realización, el punto de vista de tecnología está fuera del ámbito de la normalización.

4.2 Evaluación del ODP

Entre las ventajas del ODP para el modelado de servicios IN cabe citar:

- *El refinamiento gradual de especificaciones*: Muestra algunos posibles escenarios de refinamiento de especificaciones en ODP (véase la figura 4). Obsérvese que es posible refinar y transformar las especificaciones de manera iterativa.

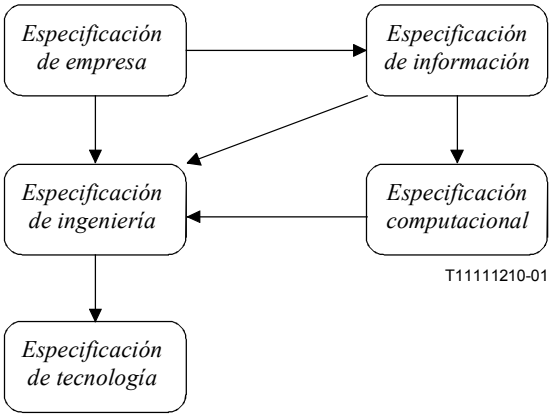


Figura 4 – Posibles escenarios de refinamiento de especificaciones en ODP

- *Mecanismos de transparencia*: El ODP proporciona diversos mecanismos de transparencia a diferentes niveles de abstracción. Estos mecanismos de transparencia comprenden: transparencia de acceso, transparencia de ubicación, transparencia de fallos, transparencia de migración y transparencia de transacciones.
- *Portabilidad de servicio y reusabilidad de componentes de servicio*: Estos dos requisitos pueden ser sustentados fácilmente en un entorno de procesamiento distribuido (DPE, *distributed processing environment*) orientado a objetos, como una plataforma basada en CORBA (*common object request broker architecture*).
- *Interacción de prestaciones*: ODP favorece decidamente el uso de técnicas de descripción formales (FDT, *formal description techniques*) en diferentes puntos de vista. El uso de FDT mejora la descripción del comportamiento del servicio y reduce considerablemente los riesgos de interacciones de servicios.
- *Funciones de denominación y comercio*: Éstas son partes de las funciones definidas por ODP. La función de comercio proporciona los medios para anunciar ofertas de servicio y descubrir ofertas de servicio a través de peticiones de servicio.

El ODP no indica ningún método para desarrollar cada punto de vista ni el lenguaje que se debe utilizar para describirlo. Por tanto, el ODP se debe aplicar junto con las técnicas de modelado apropiadas para cada uno de los puntos de vista.

4.3 Lenguaje de modelado unificado

Introducción al lenguaje de modelado unificado (UML)

El lenguaje de modelado unificado (UML, *unified modelling language*) es un lenguaje de modelado que incorpora el consenso de la comunidad orientada a objetos sobre los conceptos esenciales de modelado. Permite expresar desviaciones con respecto a sus mecanismos de extensión. Los proyectistas del UML consideraron los siguientes objetivos al elaborarlo:

- Proporcionar semántica y notación suficientes para tratar una amplia variedad de aspectos de modelado contemporáneos de manera directa y económica.
- Proporcionar semántica suficiente para tratar determinados aspectos de modelado futuros, especialmente relacionados con la tecnología de componentes, conputación distribuida, marcos y ejecutabilidad.
- Proporcionar mecanismos de extensibilidad para que cada proyecto pueda ampliar el metamodelo para su aplicación a bajo costo. No se debe obligar a los usuarios a ajustarse al metamodelo UML.

- Proporcionar mecanismos de extensibilidad para poder desarrollar métodos de modelado futuros por encima del UML
- Proporcionar semántica suficiente para facilitar el intercambio de modelos entre una variedad de clases de herramientas.
- Proporcionar semántica suficiente para especificar la interfaz con depósitos para compartir y almacenar artefactos de modelos.

4.3.1 Evaluación del UML

El UML se utiliza especialmente en los puntos de vista de información y computacional del ODP. Aunque el UML se podrá aplicar también a otros puntos de vista de ODP, actualmente la aplicación está limitada a éstos. Como un gran número de protocolos con los cuales tiene que interfuncionar la RI ya están especificados en ASN.1 y SDL, cabría considerar la compatibilidad hacia atrás para utilizar la ASN.1 y el SDL en vez del UML en el punto de vista de ingeniería.

5 Ventajas de utilizar la orientación a objetos para el modelado de servicios

El modelado orientado a objetos se basa en el principio de "objetos". Los objetos son componentes independientes, es decir, es posible describir sus propiedades prescindiendo del mundo exterior. La definición de un objeto comprende atributos y métodos. Los atributos describen los datos en los objetos; los métodos, las operaciones que pueden ser impuestas a esos datos. Los objetos pueden invocar métodos en otros objetos. Otros conceptos muy importantes del modelado orientado a objetos son la herencia y el encapsulado.

Dado el amplio ámbito y la complejidad de los servicios previstos para el CS-4 de RI (resultantes de la integración de servicios basados en redes móviles, de banda ancha e IP), las ventajas de utilizar el paradigma orientado a objetos en la normalización del CS-4 de RI son:

- Los métodos orientados a objetos parecen adecuados para especificar servicios de RI en todo el diseño. Sin embargo, se debe investigar si el modelado de servicios de RI es realmente un campo de aplicación cuando es posible aplicar también satisfactoriamente métodos OO.
- Los métodos orientados a objetos se usan ya generalizadamente en la ingeniería de soporte lógico.
- Es posible utilizar un DPE orientado a objetos como un modelo computacional uniforme que proporciona distribución transparente de la inteligencia de red (lógica de servicio, creación y gestión de servicios). Un escenario de migración podría ser el interfuncionamiento entre un DPE basado en CORBA (para la inteligencia de red) y los equipos de RI existentes (puntos de conmutación de servicios (SSP) a través de una interfaz de cabecera (SS7-IDL) como se ilustra en la figura 5:

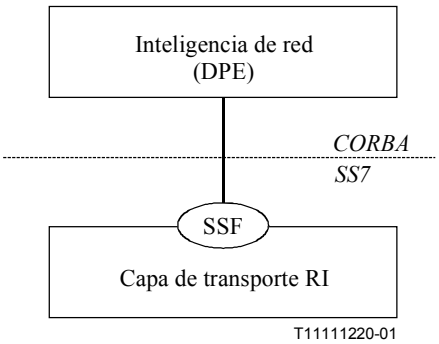


Figura 5 – Posible escenario de evolución para la RI

Entre las ventajas de utilizar un DPE orientado a objetos (por ejemplo, una plataforma basada en CORBA) para la RI cabe citar: mejora de la reutilización y generalidad de servicios (para composición y descomposición), facilidad de desarrollo, instalación e integración, vinculación dinámica y reconfiguración de componentes de servicio.

La arquitectura CORBA (*common object request broker architecture*) permite invocar métodos orientados a objetos en un entorno distribuido. Esto significa que CORBA ocultará la ubicación del objeto direccionado. Las interfaces entre objetos especificados con CORBA IDL proporcionan transparencia de tecnología, porque la interfaz ocultará cómo los objetos han sido realizados. Cabe señalar, no obstante, que actualmente una de las principales desventajas de utilizar estos entornos en el contexto de RI son los requisitos de funcionamiento en tiempo real.

La definición de interacciones entre objetos a través de una interfaz lógica suele denominarse una API. La figura 6 ilustra el uso de las API en un contexto de RI.

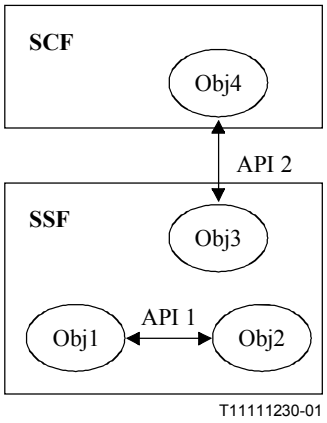


Figura 6 – Ejemplo de utilización de las API en un contexto de RI

En la figura 6, los objetos 1 y 2 residen en la SSF. El objeto 1 podría ser, por ejemplo, un objeto de modelo de llamada y el objeto 2 la tabla de estados de activación, Cuando la SCF arma un punto de detección de evento (EDP, *event detection point*) particular, el objeto 1 invocaría un método (llamada API) al objeto 2 cambiando el estado de un determinado activador. Por tanto, la llamada API podría ser "arm edp2".

Cuando el objeto 3 en la SSF invoca un método en el objeto 4 (llamada API 2) que reside en la SCF, atraviesa una interfaz externa y como resultado **puede** corresponder con una operación de protocolo. Las propiedades de la llamada API, es decir, su contenido, tendrán que efectuar la operación de protocolo física. Por consiguiente, existe una correspondencia de uno a uno entre el contenido de la llamada API y los atributos, por ejemplo, de una operación de INAP. Imagínese que el objeto 3 representa un objeto de "acceso de servicio" y el objeto 4 una lógica de servicio de RI. Al recibir información del usuario, el objeto 3 pudiera invocar un método en el objeto 4 como "recepción de cifras marcadas por el usuario". Este método podría aplicarse como la operación "AnalysedInformation" del INAP.

5.1 Investigación del uso de las API en el CS4 de RI

En el campo en evolución de las telecomunicaciones, es más importante que nunca mantenerse al día en relación con los nuevos conceptos de tecnologías y redes, particularmente en lo que respecta a la *tecnología de información*.

Una necesidad común a través de todas estas arquitecturas es la reutilización de guiones de servicio y la capacidad de enviar y recibir información empaquetada en una forma reconocible. Los

protocolos normalizados y los privados funcionan juntos y a menudo se plantea la necesidad de hacer corresponder un protocolo con otro en cabeceras o medios de interfuncionamiento.

Actualmente es mucho más evidente la necesidad de describir servicios usando un formato común. Los servicios en el dominio de las telecomunicaciones internacionales están utilizando técnicas asociadas con las API. Es posible aprovechar este trabajo en la elaboración de normas internacionales, en particular las relacionadas con la red inteligente.

5.1.1 Antecedentes

Para cualquier nueva proposición de servicio, hay que plantear dos preguntas, a saber:

1) ¿Cómo se sustentará el servicio en otras tecnologías de transporte (*interfuncionamiento de servicios-redes*)?

2) ¿Cómo interfuncionará este servicio con otros servicios (*interfuncionamiento servicio-servicio*)?

En general, no es fácil responder a estas preguntas, pero el hecho de plantearlas puede resultar en definiciones de servicio mucho más genéricas, flexibles y evolutivas. Por ejemplo, la definición de un servicio de correo vocal para la RTPC debe poder funcionar por redes basadas en IP y redes GSM, y debe poder interfuncionar con servicios de movilidad personal, correo electrónico y videotelefonía.

Se debe estudiar un método basado en componentes, que utilice API bien definidas, para especificar y construir servicios, con el fin de lograr el mejor interfuncionamiento de servicio-red y servicio-servicio.

En las subcláusulas 5.1.2 a 5.1.4 se esboza un posible marco de cómo se podría utilizar un método basado en componentes para construir servicios de telecomunicaciones con miras a especificar redes y protocolos inteligentes.

5.1.2 Marco para el uso de la API

El marco se basa en el modelo de computación de cliente/servidor. El cliente representa la entidad que solicita la ejecución de una función, el servidor representa la entidad que ejecuta la función y devuelve (facultativamente) el resultado. El cliente y el servidor pueden ser representados como componentes (conjuntos de objetos de soporte lógico o SIB). La interfaz entre el cliente y el servidor es definida por la API presentada al cliente por el servidor y es habilitada por el soporte medio subyacente (por ejemplo, protocolos de transporte).

Este método es complementario de las metodologías ya adoptadas para las normas de RI, y las mejora con una visión más orientada al soporte lógico, en particular, del plano funcional distribuido (DFP, *distributed functional plane*). Las entidades funcionales identificadas en el DFP son adecuadas para constituir los componentes iniciales para los cuales se definen las API. Por ejemplo, la información que fluye a y desde la SSF/CCF proporciona una base conveniente para definir una API.

A medida que se definan entidades más funcionales (a saber, control de conexión de portador para la RDSI-BA, mensajería vocal, corretaje de servicios), habrá que normalizar las interfaces con muchas de estas capacidades. El método basado en componentes permite que estas capacidades sean definidas por sus API, de modo que puedan ser integradas fácilmente desde la perspectiva del soporte lógico. Es posible utilizar las API como base para la elaboración de protocolos.

Para componentes complejos, éstos podrían ser separados en objetos, a partir de los cuales se definiría el protocolo. Para componentes simples, es posible derivar el protocolo directamente. Al definir la API en detalle, se simplifica la correspondencia del protocolo API, y se prepara para el futuro. Por ejemplo, será más fácil modificar las actuales normas de RI para alinearlas con tecnologías de computación.

La figura 7 muestra la relación entre servicios y componentes. Cada componente es definido por su API, y ésta refleja la funcionalidad completa del componente (incluidas la utilización gestión, etc.).

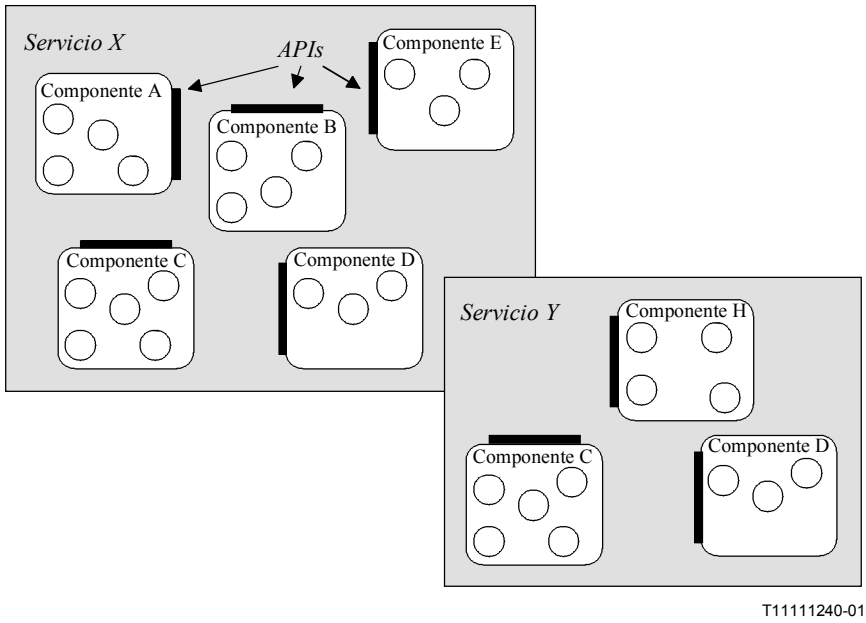


Figura 7 – Servicios/Componentes/API

5.1.3 Visión general de la API

La API es la clave para definir los componentes cliente/servidor. La API es la definición de la funcionalidad de un componente. Es esencialmente un conjunto de operaciones (o métodos) que pueden ser invocados en el componente, cada una de las cuales hace que el componente presente una funcionalidad comportamental. Cada operación es especificada sintácticamente como un identificador que identifica la operación invocada, y parámetros que afectan de alguna manera el comportamiento del componente.

Por ejemplo, la siguiente operación de API (*add_transaction*) en un componente de facturación añadiría la transacción identificada por el parámetro *transaction_id* a la factura del cliente identificado por el parámetro *customer_id*:

```
add_transaction ( transaction_id, customer_id )
```

La API es abstracta con respecto a la tecnología usada para realizar el componente y se especifica mediante un lenguaje de definición de interfaces abstractas. Además de la sintaxis, la API especifica la semántica para invocar la funcionalidad y el comportamiento del componente, a saber, cómo utilizar y modificar atributos públicos y privados, y la devolución de resultados y errores. La API define el conjunto completo de funcionalidades que un componente puede ejecutar.

Una presentación particular de una API representa la manera en que se realizan los componentes, y la sintaxis y semántica concretas para invocar una capacidad. La presentación puede restringir la funcionalidad que es accesible a un componente invocador. Un componente puede tener varias presentaciones, pero sólo tiene una API. Por ejemplo, la presentación de utilización de una API de SSF/CCF sólo incluiría las operaciones de tratamiento de llamadas, y se podría realizar utilizando protocolos INAP y/o PU-RDSI; la presentación de gestión sustentaría capacidades tales como espaciamiento de llamadas, y se podría realizar con protocolos INAP o CMIP.

5.1.4 Ejemplo de API para el procesamiento de llamada

El componente de procesamiento de llamada (SSF/CCF) tiene varios clientes potenciales, es decir, usuarios (denominados aquí partes de extremo), proveedores de servicio (denominados aquí terceros), gestión, etc. En este ejemplo sólo se consideran las partes de extremo y los terceros. La parte de extremo difiere de terceros en que es la que solicita las llamadas y conexiones portadoras que se han de establecer, mantener y liberar entre ellas. El tercero puede actuar en nombre de una parte de extremo, o modifica la manera en que funciona el procesamiento de llamada, con o sin el conocimiento de las partes de extremo.

Cada "llamada API" (operación) describe una operación efectuada por una parte de extremo o por un tercero. La llamada API será coherente entre los puntos A y B, aunque el protocolo y los mecanismos de red subyacentes pueden variar en diversos puntos entre los dos. El siguiente ejemplo (figura 8) es aclaratorio.

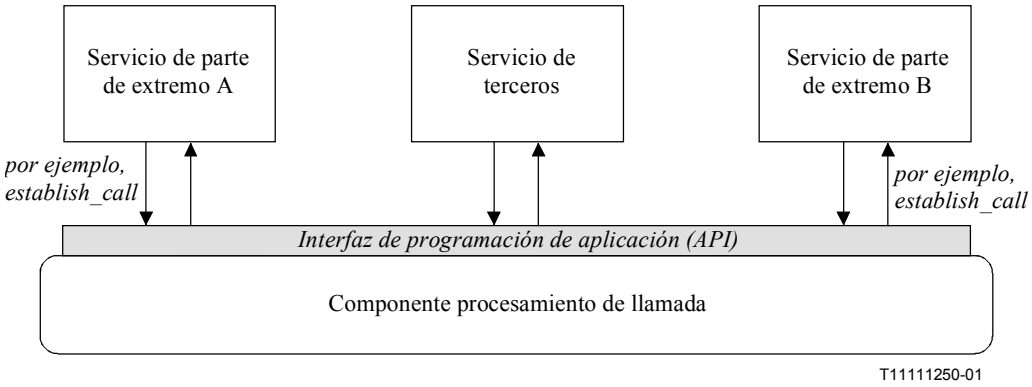


Figura 8 – Ejemplo de API de procesamiento de llamada

Se supone que un evento (descolgar un teléfono) es detectado por una parte de extremo como un servicio RDSI (que funciona en el terminal). No es probable que el terminal RDSI tenga alguna relación con la red hasta que el usuario marca un número. Cuando el servicio de la parte de extremo A ha recopilado cifras suficientes del usuario invoca la operación API "establecer llamada" en el componente "procesamiento de llamada".

Esto resultaría en el envío de la señalización necesaria a través de la red entre los diversos objetos del componente procesamiento de llamadas utilizando, por ejemplo, los protocolos DSS1 y parte usuario de la RDSI. Cuando la llamada llega al extremo distante, el componente procesamiento de llamada invocará una llamada API en un componente de servicio en el servicio de parte de extremo B, y esto se denominaría "establecer llamada", para simetría.

Si se prevé un escenario más complicado, el tercero se puede registrar con el procesamiento de llamada para la notificación de eventos cuando se cumplan los criterios específicos. Si se cumplen dichos criterios, el procesamiento de llamada invocará una operación en el componente de servicio "terceros" que solicitó la notificación.

Es bastante fácil observar cómo la API de procesamiento de llamada a un tercero puede corresponder con las normas de RI existentes (por ejemplo, INAP), pero es menos fácil la correspondencia de la API de parte de extremo, porque no ha sido aún tratada en las normas de RI.

5.2 Método de bloques de construcción independientes del servicio (SIB)

A continuación se examinan las deficiencias de la utilización del método de SIB de RI.

La división de servicios en partes reutilizables más pequeñas ha sido muy útil a la RI en el pasado, y es utilizada por muchos fabricantes como parte de sus entornos de ejecución de lógica de servicio (SLEE, *service logic execution environments*).

La idea que sustenta este método es hacer que el mayor número posible de bloques sean reutilizables y puedan ser ejecutados a través de muchas aplicaciones de servicio. El concepto de reutilización es importante y eficaz y en realidad puede aplicarse prescindiendo de cómo se modela un servicio. Sin embargo, cuando se compara la metodología de SIB con las metodologías orientadas a objetos, son evidentes las siguientes deficiencias de SIB:

- En primer lugar, es un proceso difícil de refinar (SIB) del GFP al nivel de plano funcional distribuido (DFP).
- El refinamiento del plano de servicio a SIB en el GFP sólo usa un paso intermedio, a saber, el de SIB de alto nivel. Como resultado, el refinamiento del plano de servicio al GFP no es muy preciso. Para lograr un refinamiento más gradual, el concepto de SIB de alto nivel no es suficiente, y se requiere un refinamiento más iterativo.
- El modelado del GFP por medio de los SIB no proporciona un modelo de servicio completo, porque los datos de servicio están subordinados a las funciones realizadas por un servicio RI. Los datos necesitan atención más explícita, por ejemplo, permitir una gestión mejor de los datos de servicio.
- El método SIB aplicado en el modelado del GFP de los CS1 y CS2 de RI difiere de las técnicas de modelado aplicadas en el campo de las telecomunicaciones. Sin embargo, hay una necesidad creciente de armonizar productos originados en el campo de las telecomunicaciones (por ejemplo, facturación y atención al cliente).
- Las técnicas de especificación usadas en el GFP y en el DFP no permiten mecanismos de comprobación de coherencia entre especificaciones, limitando así las composiciones y reutilización de servicios. De manera más general, no hay una metodología de especificación global preconizada por la RI.

6 Posible evolución de los SIB a capacidades de servicio orientadas a objetos

Existe una gran diferencia entre servicios y prestaciones de servicio por un lado y los flujos de información y protocolos por otro lado, lo que supone que es difícil verificar la integridad de los flujos de información y operaciones de protocolo/parámetros contra los servicios y prestaciones de servicios que tienen que ser sustentados. En los CS-1 y CS-2 de RI, el GFP se utilizó para lograr una "capa" entre los servicios y prestaciones de servicio por un lado y flujos de información y operaciones de protocolo/parámetros por otro lado. Éste es un medio de sustentar la validación de la integridad del protocolo. Para poder modelar servicios complejos, hay que mejorar las técnicas de modelado usadas actualmente en el GFP. Se considera que la orientación a objetos es una técnica prometedora para el modelado de servicios. En consecuencia, es útil investigar la evolución de las actuales técnicas de modelado a un método más orientado a objetos.

6.1 Modelo de clases de servicio

El modelo de clases de servicio comprende dos aspectos complementarios:

- *Aspecto de ejecución del servicio*. Este aspecto comprende las clases de objeto más elementales que están disponibles para el desarrollo de servicios.
- *Aspecto de provisión/adaptación del servicio*. Este aspecto muestra las clases de objeto que son visibles durante la provisión y adaptación/personalización. Oculta las clases de objeto que no pueden ser gestionadas mientras se ejecuta el servicio.

Estos dos aspectos se denominan "modelos de objetos" en varias metodologías OO, lo que significa que ambos aspectos proporcionan una visión estática del servicio. Unas clases de objeto pueden ser accedidas en la ejecución del servicio, otras en la provisión/adaptación del servicio, y algunas en ambos casos. Esto se representa en la figura 9. La clase de objeto "llamada" contiene, por ejemplo, datos que cambian durante la ejecución del servicio. La clase de objeto "anuncio" contiene datos que pueden ser leídos durante la ejecución del servicio (reproducción de un anuncio). El contenido de los datos del anuncio puede ser variado en función del sistema de gestión del servicio. Por tanto, la clase de objeto "anuncio" está en la intersección de ejecución del servicio y provisión/adaptación del servicio.

En la figura 9 se distingue entre recursos de servicio y recursos de red. Un ejemplo de la clase de objeto recurso de red es la clase de objeto terminal. La relación entre los recursos de servicio y los recursos de red indica cómo las capacidades de servicio corresponden con los recursos de red. Por ejemplo, un usuario puede hacer una llamada desde diferentes terminales, tales como su terminal de telefonía ordinaria o su terminal RDSI. Las clases de objeto de ejecución del servicio, cuyos casos son creado durante la ejecución del servicio, son *dinámicas* (a saber, llamada), todos los recursos de servicio son *persistentes,* pues su duración es mayor que una ejecución del servicio.

Para el modelado de servicios, sólo son pertinentes los recursos de servicio. Estas clases de objeto son las capacidades de servicio utilizadas para componer servicios.

Obsérvese que las clases de objeto de las figuras 9 y 10 son sólo ejemplos.

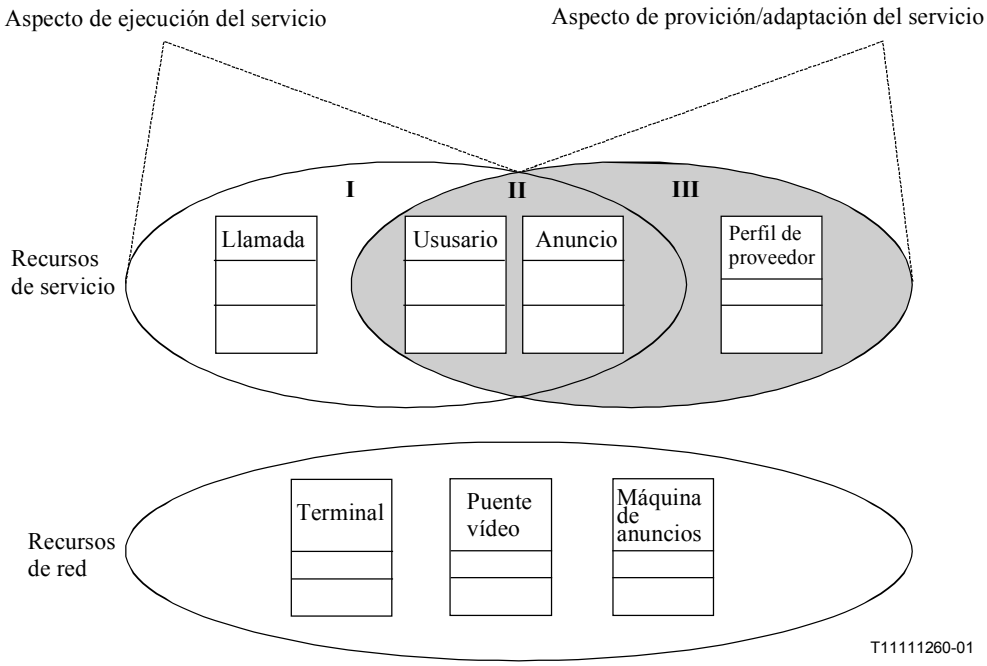


Figura 9 – Diferentes aspectos en un servicio

6.2 Aspecto de ejecución del servicio

La figura 10 muestra el modelo de objeto del aspecto de ejecución del servicio en el más alto nivel. La clase de objeto *Service Logic* representa el servicio y puede consistir en ninguna o más (0+) clases de objeto *Service Logic* y ninguna o más clases de objeto *ServiceResource.* Esto permite modelar un servicio como una composición de prestaciones de servicio (más general, permite el refinamiento gradual). En el nivel más bajo de descomposición, figuran las clases de objeto *ServiceResource* que tienen un carácter independiente del servicio y pueden ser consideradas el

equivalente orientado a objeto de los SIB usados actualmente. Una clase de objeto *ServiceResource* puede ser reutilizada en ninguna o más clases de objeto *Service Logic* (0+). Un caso de la clase de objeto *ServiceResource* es un *DynServResource* o un *PersServResource*. La clase de objeto *DynServResource* representa todas las clases de objeto cuyos objetos existen sólo durante una ejecución del servicio (véase I en la figura 9). La clase de objeto *PersServResource* representa todas las clases cuyos objetos son persistentes, es decir, su ciclo de vida es más largo que una ejecución del servicio, por ejemplo, el periodo de tiempo que un cliente está abonado a un servicio (véase II en la figura 9). La clase de objeto *ServSessionMgr* invoca el servicio y podría ser considerada como el proceso de llamada básica en el GFP del CS-2 de RI. *ServSessionMgr* puede ser considerada por sí misma como una especialización de la clase de objeto *DynServResource*, pero se muestra separadamente debido a su cometido especial.

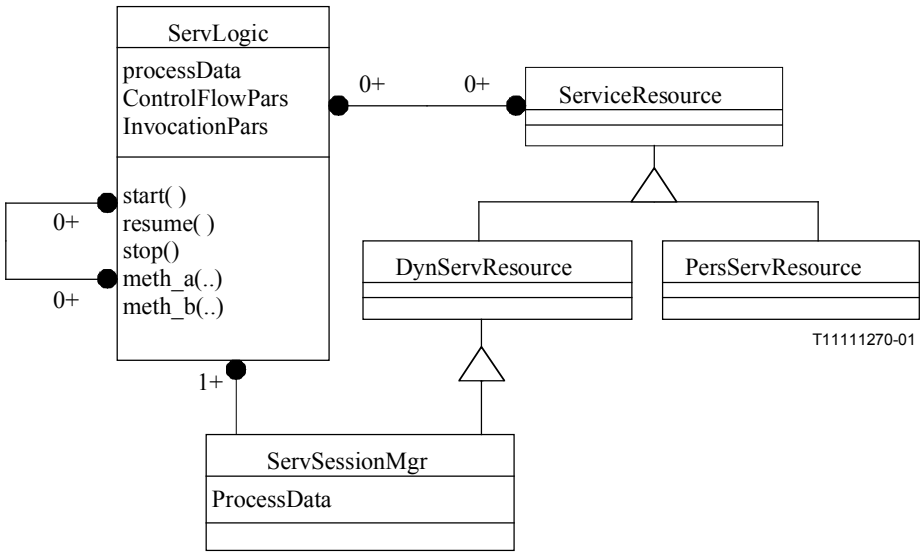


Figura 10 – Modelo de objetos del aspecto de ejecución del servicio

Cada una de las clases de objeto *DynServResource* y *PersServResource* pueden tener ninguna o más especializaciones. Una especialización de la clase de objeto *DynServResource* podría ser la clase de objeto *Call* (véase la figura 11). Un objeto de esta clase puede contener, por ejemplo, datos de una llamada, como el número de A y el número de B. Los métodos de esta clase podrán ser Add_Party( ), Remove_Party( ) y Connect( ). Posibles especializaciones de PersServResource son *Announcement* y *Counter* . Estas clases de objeto son persistentes porque existen después de una ejecución del servicio. Se consideran en los aspectos de ejecución de servicio y de provisión/adaptación de servicio, porque es posible invocar métodos en objeto de estas clases en ambos aspectos. Por ejemplo, durante la ejecución de la lógica de servicio, los datos de un objeto *Announcement* pueden ser leídos y durante la adaptación del servicio, el texto del anuncio puede ser modificado.

6.3 Migración de los SIB del CS-2 a clases de objeto y métodos

Con el fin de lograr una evolución de las técnicas de modelado de servicios actualmente utilizadas hacia un método más orientado a objetos, es útil hacer corresponder los SIB con clases de objeto y métodos de objeto. La correspondencia presentada en el cuadro que sigue a la figura 11 muestra cómo se podría hacer esto.

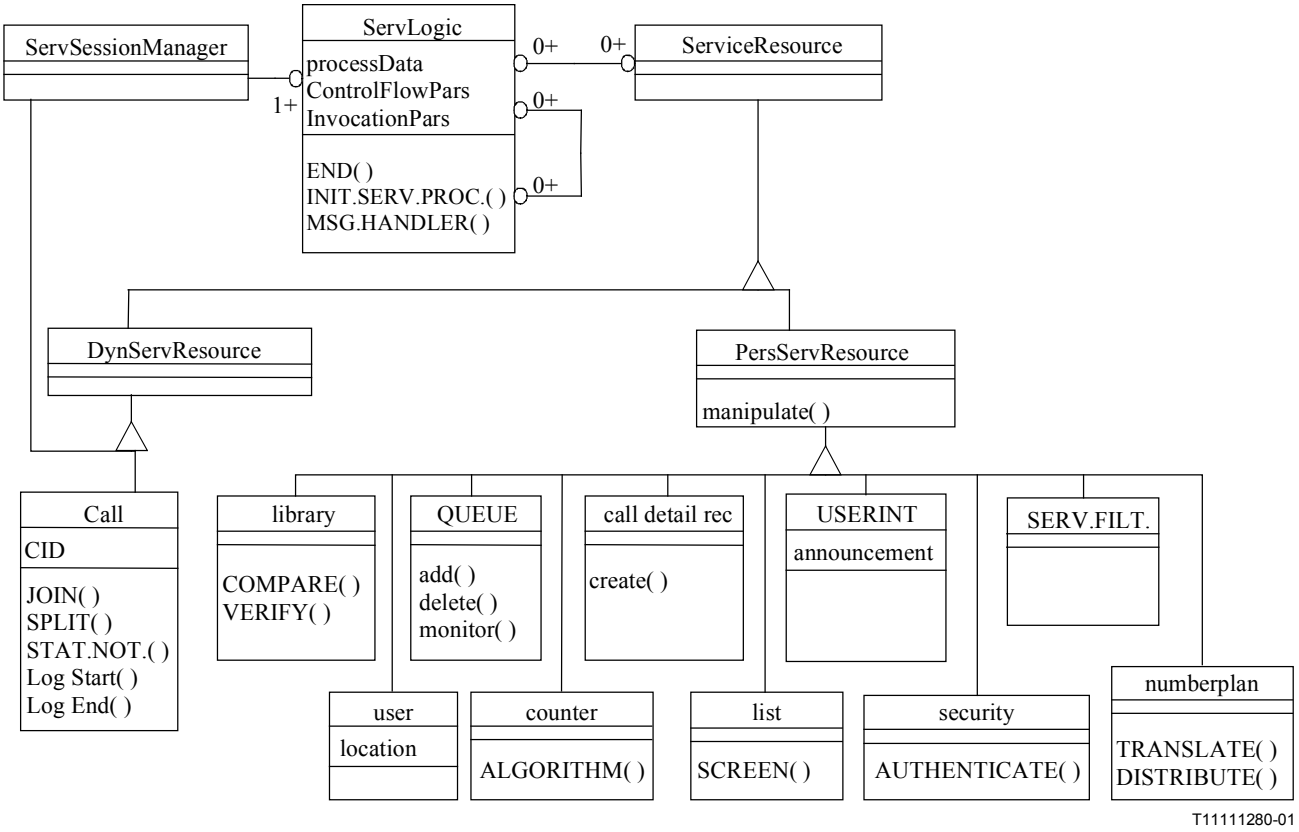


Figura 11 – Posible migración de los SIB a clases de objeto y métodos de objeto

El siguiente cuadro describe la correspondencia de los SIB del CS-2 de RI (Recomendación Q.1223) con las clases de objeto utilizadas anteriormente.

SIB	Correspondencia
Algoritmo	Aplica un algoritmo matemático a un valor (un incremento o decremento de un contador) por lo que se puede considerar como un método de la clase de objeto *Counter*
Autenticación	Función con fines de seguridad, por lo que se puede considerar como un método de la clase de objeto *Security*
Tasación	La tasación puede considerarse como la creación de un registro de detalles de llamada, y se puede representar como el método *Create* en la clase de objeto *Call Detail Record*
Comparación	Función utilizada para comprobar un identificador contra un valor de referencia especificado, tal como hora del día, lugar de origen o valor de cifras. Se puede considerar como una operación matemática genérica, por lo que se representa como un método en la clase de objeto *Library* (que puede ser considerada como una biblioteca estándar con operaciones matemáticas y de cadenas como ocurre en muchos lenguajes de programación
Distribución	La distribución de una llamada a un destino de acuerdo con un algoritmo particular se puede considerar un método en la clase de objeto *Numberplan*
Fin	Indica la terminación normal de un proceso de servicio y se puede considerar como un método de la clases de objeto *Service Logic*
Inicio de proceso de servicio	La invocación de lógica de servicio paralela se puede considerar como un método de la clase de objeto del más alto nivel, es decir, la clase de objeto *Service Logic*

SIB	Correspondencia
Incorporación	La anexión de una parte o partes en la llamada de un grupo de llamadas vigentes a otro grupo de la misma llamada se puede considerar como un método de la clase de objeto *Call*
Registro de información de llamada	El registro de los datos de la llamada identificada se considera como los métodos *Log Start* y *Log End* en la clase de objeto *Call*, en la cual los datos de esa llamada son atributos
Manejador de mensajes	La comunicación entre procesos en un solo servicio se puede considerar como un método de la clase de objeto de más alto nivel, es decir, la clase de objeto *Service Logic*
Cola	Un objeto *Queue* contiene ninguna o más referencias a objetos *Call*. Es posible añadir, suprimir, etc., casos de *Call* a/de *Queue*
Cribado	La comprobación de un número contra una lista de otros números (por ejemplo, para cribado de llamadas) se puede considerar un método de la clase de objeto *List*
Gestión de datos de servicio	Ejecuta operaciones en datos de servicio, tales como añadir, suprimir, modificar, extraer, etc. Estas operaciones son recogidas en la clase de objeto *PersServResource* como el método *Manipulate*
Filtro de servicios	Filtra llamadas de acuerdo con un mecanismo específico. Se considera una clase de objeto separada con métodos tales como *Activate* y *Report*
Separación	Separa una parte o grupo de partes en la llamada de la llamada vigente y anexa la parte/partes a una llamada nueva o ya existente. Podría ser un método de la clase de objeto *Call*
Notificación de estado	Proporciona la capacidad de indagar el estado (cambios) de recursos de red. Se puede considerar como un método en la clase de objeto *Call*, que permite verificar el estado de los recursos de red que hacen posible *Call*
Traducción	La traducción de un número a otro podría ser un método en la clase de objeto *Numberplan*
Interacción de usuario	El mecanismo de aviso y recopilación para proporcionar información al usuario y obtenerla de éste se considera una clase de objeto separada
Verificación	Puede admitir la comprobación de sintaxis de todas las clases de datos y se representa como un método de la clase de objeto *Library*

APÉNDICE I

Bibliografía

[TINA] TINA-C architecture especifications (e.g. Service Architecture and Information Architecture)

[OMT] Object-Oriented Modelling and Design, *Rumbaugh y otros*.

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 Y	Infraestructura mundial de la información y aspectos del protocolo Internet
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Impreso en Suiza
Ginebra, 2000