



UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T

X.903

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

(11/95)

**REDES DE DATOS Y COMUNICACIÓN
ENTRE SISTEMAS ABIERTOS
PROCESAMIENTO DISTRIBUIDO ABIERTO**

**TECNOLOGÍA DE LA INFORMACIÓN –
PROCESAMIENTO DISTRIBUIDO ABIERTO –
MODELO DE REFERENCIA: ARQUITECTURA**

Recomendación UIT-T X.903

(Anteriormente «Recomendación del CCITT»)

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. En el UIT-T, que es la entidad que establece normas mundiales (Recomendaciones) sobre las telecomunicaciones, participan unos 179 países miembros, 84 empresas de explotación de telecomunicaciones, 145 organizaciones científicas e industriales y 38 organizaciones internacionales.

Las Recomendaciones las aprueban los Miembros del UIT-T de acuerdo con el procedimiento establecido en la Resolución N.º 1 de la CMNT (Helsinki, 1993). Adicionalmente, la Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), que se celebra cada cuatro años, aprueba las Recomendaciones que para ello se le sometan y establece el programa de estudios para el periodo siguiente.

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. El texto de la Recomendación UIT-T X.903 se aprobó el 21 de noviembre de 1995. Su texto se publica también, en forma idéntica, como Norma Internacional ISO/CEI 10746-3.

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.

© UIT 1997

Es propiedad. Ninguna parte de esta publicación puede reproducirse o utilizarse, de ninguna forma o por ningún medio, sea éste electrónico o mecánico, de fotocopia o de microfilm, sin previa autorización escrita por parte de la UIT.

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

(Febrero de 1994)

ORGANIZACIÓN DE LAS RECOMENDACIONES DE LA SERIE X

Dominio	Recomendaciones
REDES PÚBLICAS DE DATOS	
Servicios y facilidades	X.1-X.19
Interfaces	X.20-X.49
Transmisión, señalización y conmutación	X.50-X.89
Aspectos de redes	X.90-X.149
Mantenimiento	X.150-X.179
Disposiciones administrativas	X.180-X.199
INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Modelo y notación	X.200-X.209
Definiciones de los servicios	X.210-X.219
Especificaciones de los protocolos en modo conexión	X.220-X.229
Especificaciones de los protocolos en modo sin conexión	X.230-X.239
Formularios para enunciados de conformidad de implementación de protocolo	X.240-X.259
Identificación de protocolos	X.260-X.269
Protocolos de seguridad	X.270-X.279
Objetos gestionados de capa	X.280-X.289
Pruebas de conformidad	X.290-X.299
INTERFUNCIONAMIENTO ENTRE REDES	
Generalidades	X.300-X.349
Sistemas móviles de transmisión de datos	X.350-X.369
Gestión	X.370-X.399
SISTEMAS DE TRATAMIENTO DE MENSAJES	X.400-X.499
DIRECTORIO	X.500-X.599
GESTIÓN DE REDES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS Y ASPECTOS DE SISTEMAS	
Gestión de redes	X.600-X.649
Denominación, direccionamiento y registro	X.650-X.679
Notación de sintaxis abstracta uno	X.680-X.699
GESTIÓN DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	X.700-X.799
SEGURIDAD	X.800-X.849
APLICACIONES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Cometimiento, concurrencia y recuperación	X.850-X.859
Tratamiento de transacciones	X.860-X.879
Operaciones a distancia	X.880-X.899
PROCESAMIENTO DISTRIBUIDO ABIERTO	X.900-X.999

ÍNDICE

		<i>Página</i>
Resumen		v
Introducción.....		v
1 Alcance.....		1
2 Referencias normativas		1
2.1 Recomendaciones Normas Internacionales idénticas.....		1
2.2 Pares de Recomendaciones Normas Internacionales de contenido técnico equivalente		2
3 Definiciones		2
3.1 Definiciones descriptivas		2
3.2 Abreviaturas.....		3
4 Marco		3
4.1 Puntos de vista		3
4.1.1 Conceptos.....		3
4.1.2 Utilización de puntos de vista		4
4.2 Lenguajes de punto de vista ODP		4
4.2.1 Concepto		4
4.2.2 Utilización de lenguajes de punto de vista.....		5
4.3 Funciones ODP		5
4.3.2 Utilización de funciones ODP.....		5
4.4 Transparencias de distribución ODP.....		5
4.4.1 Conceptos.....		5
4.4.2 Utilización de la transparencia de distribución		6
4.5 Normas derivadas del marco.....		6
4.6 Conformidad		7
5 Lenguaje de empresa.....		7
5.1 Conceptos.....		7
5.2 Reglas de estructuración		7
5.3 Conformidad y puntos de referencia		8
6 Lenguaje de información.....		8
6.1 Conceptos.....		9
6.2 Reglas de estructuración		9
6.3 Conformidad y puntos de referencia		9
7 Lenguaje computacional		10
7.1 Conceptos.....		10
7.2 Reglas de estructuración		11
7.2.1 Reglas de denominación		12
7.2.2 Reglas de interacción		12
7.2.2.1 Reglas de interacción de señales		13
7.2.2.2 Reglas de interacción de trenes.....		13
7.2.2.3 Reglas de interacción de operaciones		13
7.2.2.4 Reglas de parámetros		13
7.2.2.5 Flujos, operaciones y señales		13
7.2.3 Reglas de vinculación		14
7.2.3.1 Reglas de vinculación implícita para interfaces de operaciones de servidor ..		14
7.2.3.2 Reglas de vinculación primitiva.....		14
7.2.3.3 Reglas para la vinculación compuesta		14
7.2.4 Reglas de tipos		15
7.2.4.1 Reglas de subtipificación de firmas de interfaz de señales		15
7.2.4.2 Reglas de subtipificación de firmas de interfaz de trenes		16
7.2.4.3 Reglas de subtipificación de firmas de interfaz de operaciones.....		16

	7.2.5	Reglas de plantillas	16
	7.2.5.1	Reglas de plantillas de objeto computacional	16
	7.2.5.2	Instanciación de interfaz computacional	17
	7.2.5.3	Instanciación de plantilla de objeto computacional	17
	7.2.6	Reglas de fallos	17
	7.2.7	Reglas de portabilidad.....	17
	7.3	Conformidad y puntos de referencia	18
8		Lenguaje de ingeniería	18
	8.1	Conceptos.....	18
	8.2	Reglas de estructuración	20
	8.2.1	Reglas de canales	20
	8.2.1.1	Stubs.....	21
	8.2.1.2	Vinculadores	23
	8.2.1.3	Objetos de protocolo	23
	8.2.1.4	Interceptores.....	23
	8.2.2	Reglas de referencia de interfaz	23
	8.2.3	Reglas de vinculación distribuida	24
	8.2.4	Reglas de reubicación	25
	8.2.5	Reglas de conglomerados.....	25
	8.2.6	Reglas de cápsulas	26
	8.2.7	Reglas de nodos	27
	8.2.8	Reglas de gestión de aplicaciones.....	28
	8.2.9	Reglas de fallos	29
	8.3	Conformidad y puntos de referencia.....	29
9		Lenguaje de tecnología	30
	9.1	Conceptos.....	30
	9.2	Reglas de estructuración	30
	9.3	Conformidad y puntos de referencia.....	30
10		Reglas de consistencia (o coherencia).....	30
	10.1	Correspondencias de especificaciones computacionales y de especificaciones de información	31
	10.2	Correspondencias de especificaciones de ingeniería y de especificaciones computacionales.....	31
11		Funciones ODP	32
12		Funciones de gestión	33
	12.1	Función de gestión de nodo	33
	12.1.1	Gestión de hilos.....	33
	12.1.2	Acceso a relojes y gestión de temporizadores.....	34
	12.1.3	Creación de canales y ubicación de interfaces	34
	12.1.4	Instanciación de plantilla de cápsula y supresión de cápsula.....	34
	12.2	Función de gestión de objeto	34
	12.3	Función de gestión de conglomerado.....	35
	12.3.1	Punto de comprobación de conglomerado	35
	12.3.2	Supresión, desactivación y fallo de conglomerado	35
	12.3.3	Reactivación y recuperación de conglomerado.....	36
	12.3.4	Traslado de conglomerado	36
	12.4	Función de gestión de cápsula	36
	12.4.1	Instanciación de plantilla de conglomerado.....	36
	12.4.2	Supresión de cápsula.....	36
13		Funciones de coordinación.....	37
	13.1	Función de notificación de evento	37
	13.1.1	Conceptos.....	37
	13.1.2	Reglas.....	37

13.2	Función de punto de comprobación y recuperación	37
13.2.1	Verificación por punto de comprobación.....	37
13.2.2	Recuperación.....	38
13.3	Función de desactivación y reactivación.....	38
13.3.1	Desactivación.....	38
13.3.2	Reactivación.....	39
13.4	Función de grupo	39
13.4.1	Conceptos.....	39
13.4.2	Reglas.....	39
13.5	Función de replicación	39
13.6	Función de traslado	39
13.6.1	Replicación	40
13.6.2	Desactivación y reactivación	40
13.7	Función de transacción	40
13.7.1	Conceptos.....	40
13.7.2	Reglas.....	40
13.8	Función de transacción ACID.....	40
13.9	Función de rastreo de referencia de interfaz de ingeniería	41
14	Funciones de depositario	41
14.1	Función de almacenamiento.....	41
14.1.1	Conceptos.....	41
14.1.2	Reglas.....	42
14.2	Función de organización de información	42
14.3	Función de reubicación	42
14.3.1	Conceptos.....	42
14.3.2	Reglas.....	42
14.4	Función de depositario de tipos	43
14.4.1	Reglas.....	43
14.5	Función de comercio.....	43
14.5.1	Conceptos.....	43
14.5.2	Reglas.....	44
15	Funciones de seguridad	44
15.1	Conceptos.....	44
15.2	Función de control de acceso	44
15.3	Función de auditoría de seguridad	44
15.4	Función de autenticación	45
15.5	Función de integridad	45
15.6	Función de confidencialidad	46
15.7	Función de no repudio	46
15.8	Función de gestión de claves	46
16	Transparencia de distribución ODP	47
16.1	Transparencia de acceso	48
16.2	Transparencia de fallo.....	48
16.2.1	Conceptos.....	48
16.2.2	Reglas.....	48
16.2.2.1	Replicación	48
16.2.2.2	Punto de comprobación y recuperación	48
16.3	Transparencia de ubicación.....	48
16.4	Transparencia de traslado	48
16.4.1	Conceptos.....	49
16.4.2	Reglas.....	49
16.5	Transparencia de persistencia	49
16.5.1	Conceptos.....	49
16.5.2	Reglas.....	49
16.6	Transparencia de reubicación.....	49

	<i>Página</i>
16.7	Transparencia de replicación 50
16.7.1	Conceptos..... 50
16.7.2	Reglas..... 50
16.8	Transparencia de transacción 50
16.8.1	Concepto 50
16.8.2	Reglas..... 50
Anexo A	– Reglas formales para el establecimiento de supertipos/subtipos de interfaces computacionales..... 51
A.1	Notaciones y convenios 51
A.2	Sistema de tipos 51
A.2.1	Reglas de tipificación..... 52
A.2.2	Definiciones de tipos..... 52
A.2.3	Un algoritmo para la comprobación de tipos 53
A.3	Tipos de firma de interfaz de señales..... 54
A.4	Tipos de firma de interfaz de operaciones 55
A.5	Tipos de interfaz de trenes 55
A.6	Ejemplo..... 56

Resumen

Esta Recomendación | Norma Internacional contiene la especificación de las características requeridas que califican los sistemas de procesamiento distribuido como abiertos. Éstas son las constricciones que deben cumplir las normas de procesamiento distribuido abierto (ODP). Se emplean las técnicas descriptivas de la Recomendación X.902.

Introducción

El rápido crecimiento del procesamiento distribuido ha creado la necesidad de un marco de coordinación para la normalización del procesamiento distribuido abierto (ODP, *open distributed processing*). Este modelo de referencia de ODP proporciona tal marco. Crea una arquitectura dentro de la cual se puede integrar un soporte de distribución, interfuncionamiento y portabilidad.

El modelo de referencia de procesamiento distribuido abierto (RM-ODP), Recs. UIT-T X.901 a X.904 | ISO/CEI 10746, se basa en conceptos precisos utilizados en trabajos actualmente en curso para el desarrollo del procesamiento distribuido y, en la medida de lo posible, en el empleo de técnicas de descripción formal para la especificación de la arquitectura.

El modelo de referencia de procesamiento distribuido abierto está constituido por

- Rec. UIT-T X.901 | ISO/CEI 10746-1: **Visión de conjunto**: contiene una visión de conjunto de las motivaciones del ODP, que da el alcance, la justificación y la explicación de conceptos esenciales, y una descripción de la arquitectura ODP. Contiene material explicativo sobre la interpretación y aplicación del RM-ODP por los usuarios, que pueden incluir escritores de normas y arquitectos de sistemas ODP. Contiene también una agrupación en categorías de las áreas requeridas de normalización, expresadas en términos de los puntos de referencia para conformidad identificados en esta Recomendación | Norma Internacional. Esta parte no es normativa.
- Rec. UIT-T X.902 | Norma ISO/CEI 10746-2: **Fundamentos**: contiene la definición de los conceptos y marco analítico para la descripción normalizada de sistemas de procesamiento distribuido (arbitrarios). Introduce los principios de conformidad con las normas ODP y la forma en que deben aplicarse. La exposición se hace solamente a un nivel de detalle suficiente para la aplicación de esta Recomendación | Norma Internacional y el establecimiento de los requisitos que cumplirán las nuevas técnicas de especificación. Esta parte es normativa.
- Rec. UIT-T X.903 | ISO/CEI 10746-3: **Arquitectura**: contiene la especificación de las características que debe tener un procesamiento distribuido para que sea abierto. Estas son las constricciones a que deben ajustarse las normas ODP. Emplea las técnicas descriptivas de la Rec. UIT-T X.902 | ISO/CEI 10746-2. Esta parte es normativa.
- Rec. UIT-T X.904 | ISO/CEI 10746-4: **Semántica arquitectural**: contiene una formalización de los conceptos de modelado ODP definidos en la Rec. UIT-T X.902 | ISO/CEI 10746-2, cláusulas 8 y 9. La formalización se consigue interpretando cada concepto en término de las construcciones de las diferentes técnicas de descripción formal normalizadas. Esta parte es normativa.

Esta Recomendación | Norma Internacional contiene un anexo (este anexo es parte integrante del modelo de referencia).

NORMA INTERNACIONAL

RECOMENDACIÓN UIT-T

TECNOLOGÍA DE LA INFORMACIÓN – PROCESAMIENTO DISTRIBUIDO ABIERTO – MODELO DE REFERENCIA: ARQUITECTURA

1 Alcance

Esta Recomendación UIT-T | Norma internacional:

- define la manera de especificar los sistemas ODP, utilizando conceptos presentados en la Recomendación UIT-T X.902 | ISO/CEI 10746-2;
- identifica las características que califican los sistemas como sistemas ODP.

Establece un marco para coordinar el desarrollo de normas existentes y futuras para los sistemas ODP y puede servir de referencia en esas normas.

2 Referencias normativas

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

2.1 Recomendaciones | Normas Internacionales idénticas

- Recomendación UIT-T X.200 (1994) | ISO/CEI 7498-1:1994, *Tecnología de la información – Interconexión de sistemas abiertos – Modelo de referencia básico: El modelo básico.*
- Recomendación UIT-T X.810 (1995) | ISO/CEI 10181-1:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Visión general.*
- Recomendación UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de autenticación.*
- Recomendación UIT-T X.812 (1995) | ISO/CEI 10181-3:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de control de acceso.*
- Recomendación UIT-T X.813¹⁾ | ISO/CEI 10181-4...¹⁾, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de no repudio.*
- Recomendación UIT-T X.814 (1995) | ISO/CEI 10181-5:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de confidencialidad.*
- Recomendación UIT-T X.815 (1995) | ISO/CEI 10181-6:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de integridad.*
- Recomendación UIT-T X.816 (1995) | ISO/CEI 10181-7:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de auditoría y alarmas de seguridad.*
- Recomendación UIT-T X.902 (1995) | ISO/CEI 10746-2:1996, *Tecnología de la información – Procesamiento distribuido abierto – Modelo de referencia: Fundamentos.*

¹⁾ Actualmente en estado de proyecto.

2.2 Pares de Recomendaciones | Normas Internacionales de contenido técnico equivalente

- Recomendación X.800 del CCITT (1991), *Arquitectura de seguridad de interconexión de sistemas abiertos para aplicaciones del CCITT*.
ISO 7498-2:1989, *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture*.

3 Definiciones

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

3.1 Definiciones descriptivas

Este modelo de referencia utiliza el siguiente término definido en la Rec. UIT-T X.200 | ISO/CEI 7498-1:

- sintaxis de transferencia.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.811 | ISO/CEI 10181-2:

- declarante (o reclamante);
- información de autenticación de intercambio;
- principal;
- tercero de confianza.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.812 | ISO/CEI 10181-3:

- información de control de acceso;
- función de decisión de acceso;
- función de ejecución de acceso (o función de cumplimiento de acceso);
- iniciador;
- objetivo (o destino).

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.813 | ISO/CEI 10181-4:

- generador de evidencia (o de prueba);
- usuario de evidencia (o de prueba);
- verificador de evidencia (o de prueba);
- originador (de datos no repudiables);
- recipiente (de datos no repudiables);
- prueba de no repudio;
- solicitante del servicio de no repudio;
- notario.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.814 | ISO/CEI 10181-5:

- datos de confidencialidad protegida;
- ocultación, ocultar;
- originador;
- recipiente;
- revelación, revelar.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.815 | ISO/CEI 10181-6:

- datos de integridad protegida;
- originador;

- recipiente;
- protección;
- validación, validar.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.816 | ISO/CEI 10181-7:

- función (de) colector de alarmas;
- función (de) examinador de alarma;
- función (de) analizador de rastreo (o de pista) de auditoría;
- función (de) examinador de rastreo (o de pista) de auditoría;
- función (de) archivador de rastreo (o de pista) de auditoría;
- función (de) registrador de auditoría;
- función (de) colector de rastreo (o de pista) de auditoría.

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en ISO/CEI 11170-1, Key Management Framework (marco de gestión de claves):

- generación de claves;
- registro de claves;
- certificación de claves;
- desregistro de claves (o anulación de registro de claves);
- distribución de claves;
- almacenamiento de claves;
- archivo de claves;
- supresión de claves.

Este modelo de referencia utiliza los términos definidos en la Rec. UIT-T X.902 | ISO/CEI 10746-2 indicados en la tabla de la Figura 1.

3.2 Abreviaturas

A los efectos de esta Recomendación | Norma Internacional se utilizan las siguientes abreviaturas:

- | | |
|-----|---|
| ODP | Procesamiento distribuido abierto (<i>open distributed processing</i>). |
| OSI | Interconexión de sistemas abiertos (<i>open systems interconnection</i>). |

4 Marco

Este modelo de referencia define un marco que comprende:

- cinco *puntos de vista*, denominados puntos de vista de empresa, de información, computacional, de ingeniería y de tecnología, que sirven de base para la especificación de sistemas ODP;
- un *lenguaje de punto de vista* para cada punto de vista, que define conceptos y reglas para la especificación de sistemas ODP desde el punto de vista correspondiente;
- especificaciones de las *funciones* requeridas para soportar sistemas ODP;
- *prescripciones de transparencia* que muestran la utilización de las funciones ODP para lograr la transparencia de distribución.

La arquitectura para los sistemas ODP y la composición de funciones se determinan por la combinación del lenguaje computacional, el lenguaje de ingeniería y las prescripciones de transparencia.

4.1 Puntos de vista

4.1.1 Conceptos

4.1.1.1 Punto de vista de la empresa: Punto de vista de un sistema ODP y su entorno que destaca principalmente la finalidad, el alcance y las políticas de ese sistema.

abstracción	falta (o avería, o defecto)	política
acción de ramificación permanente	fallo	procesamiento distribuido abierto
acción	firma de interfaz	procesamiento distribuido
actividad	gestión de comunicaciones	prohibición
arquitectura	grupo <x>	punto de vista.
atomicidad	hilo	punto de referencia
calidad de servicio	identificador	punto de referencia de interfuncionamiento
clase	información	punto de referencia programática
comercio	instancia	punto de conformidad
comportamiento de establecimiento	instanciación	punto de referencia de intercambio
comportamiento	interacción	punto de referencia perceptual
composición	interfaz	refinamiento
comunicación	introducción	papel (o rol)
configuración	invariante	sistema ODP
contexto de denominación	nombre	sistema
contrato	normas ODP	subdominio
creación	notificación	subtipo
datos	objeto productor	supertipo
descomposición	objeto cliente	supresión
dominio <x>	objeto servidor	término
dominio de denominación	objeto iniciador	tipo
enlace ("liaison")	objeto consumidor	transparencia de distribución
entidad	objeto	ubicación en el tiempo
entorno	obligación	ubicación en el espacio
error	permiso	vinculación
estabilidad	persistencia	
estado	plantilla <x>	

Figura 1 – Términos tomados de la Rec. UIT-T X.902 | ISO/CEI 10746-2

4.1.1.2 Punto de vista de la información: Punto de vista de un sistema ODP y su entorno que destaca la semántica de la información y el procesamiento de la información.

4.1.1.3 Punto de vista computacional: Punto de vista de un sistema ODP y su entorno que permite la distribución mediante una descomposición funcional del sistema en objetos que interactúan en interfaces.

4.1.1.4 Punto de vista de la ingeniería: Punto de vista de un sistema ODP y su entorno que destaca los mecanismos y funciones requeridos para soportar la interacción distribuida entre objetos en el sistema.

4.1.1.5 Punto de vista de la tecnología: Punto de vista de un sistema ODP y su entorno que destaca la elección de la tecnología para ese sistema.

4.1.2 Utilización de puntos de vista

Los puntos de vista de la empresa, la información, computacional, la ingeniería y la tecnología se han elegido como un conjunto necesario y suficiente para satisfacer las necesidades de normas ODP. Los puntos de vista pueden aplicarse en un nivel apropiado de abstracción, a un sistema completo ODP, en cuyo caso el entorno define el contexto en el que funciona el sistema ODP. Los puntos de vista pueden también aplicarse a componentes individuales de un sistema ODP, en cuyo caso el entorno del componente incluirá alguna abstracción del entorno del sistema y otros componentes del sistema.

NOTA – El proceso de abstracción pudiera ser tal que el entorno del sistema y los otros componentes se reunieran y formarían un solo objeto.

4.2 Lenguajes de punto de vista ODP

4.2.1 Concepto

4.2.1.1 Lenguaje <Viewpoint>: Definiciones de conceptos y reglas para la aplicación de un sistema ODP a partir del punto de vista <viewpoint>; por tanto, **lenguaje de ingeniería:** definiciones de conceptos y reglas para la especificación de un sistema ODP a partir del punto de vista de ingeniería.

4.2.2 Utilización de lenguajes de punto de vista

Este modelo de referencia define un conjunto de cinco lenguajes, cada uno de los cuales corresponde a uno de los puntos de vista definidos en 4.1.1. Cada lenguaje se utiliza para la especificación de un sistema ODP a partir del punto de vista correspondiente. Estos lenguajes son:

- el lenguaje de empresa (definido en la cláusula 5);
- el lenguaje de información (definido en la cláusula 6);
- el lenguaje computacional (definido en la cláusula 7);
- el lenguaje de ingeniería (definido en la cláusula 8);
- el lenguaje de tecnología (definido en la cláusula 9).

Cada lenguaje emplea conceptos tomados de la Rec. UIT-T X.902 | ISO/CEI 10746-2, e introduce refinamientos de los conceptos, reglas prescriptivas y conceptos específicos de punto de vista adicionales relevantes para la naturaleza de la especificación en cuestión. Estos conceptos adicionales se definen, a su vez, utilizando conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2.

Una especificación de sistema comprende una o más especificaciones de punto de vista. Estas especificaciones tienen que ser mutuamente coherentes. Las reglas para la estructuración coherente de especificaciones de punto de vista se dan en la cláusula 10. El especificador tiene que demostrar por otros medios que los términos de la especificación se utilizan coherentemente. Por lo general, una especificación de un sistema mediante un determinado número de especificaciones de puntos de vista restringirá más las implementaciones que una especificación hecha mediante un menor número de especificaciones de puntos de vista. Los objetos especificados en un punto de vista pueden especificarse utilizando el lenguaje de punto de vista asociado a ese punto de vista, o utilizando los lenguajes de punto de vista asociados con otros puntos de vista. No es necesario especificar un objeto completamente desde todos los puntos de vista para obtener un conjunto mutuamente coherente de especificaciones de puntos de vista.

NOTAS

- 1 La lista de términos tomados de la Rec. UIT-T X.902 | ISO/CEI 10746-2 se enumeran en la Figura 1.
- 2 Cuando un término de la Rec. UIT-T X.902 | ISO/CEI 10746-2 es calificado por el nombre de un punto de vista (por ejemplo, «objeto **computacional**») habrá de interpretarse que dicho término de la Rec. UIT-T X.902 | ISO/CEI 10746-2 se utiliza sujeto a cualquier disposición adicional que se especifique en el lenguaje de punto de vista indicado.
- 3 Cuando un término de la Rec. UIT-T X.902 | ISO/CEI 10746-2 se utiliza sin ningún calificativo en una especificación de punto de vista (por ejemplo, «interfaz») habrá de interpretarse como si el término estuviera calificado por el nombre del punto de vista (es decir, «interfaz computacional»), si el lenguaje de punto de vista asociado impone constricciones adicionales al término.

4.3 Funciones ODP

4.3.1 Función ODP: Función requerida para soportar el procesamiento distribuido abierto.

4.3.2 Utilización de funciones ODP

Este modelo de referencia especifica, en sus cláusulas 11 a 15, las funciones requeridas para obtener el procesamiento distribuido abierto.

Cada descripción de función ODP contiene:

- una explicación del uso de la función para el procesamiento distribuido abierto;
- enunciados prescriptivos, sobre la estructura y comportamiento de la función, suficientes para asegurar la integridad global del modelo de referencia;
- un enunciado de otras funciones ODP de las que depende.

4.4 Transparencias de distribución ODP

4.4.1 Conceptos

4.4.1.1 Transparencia de acceso: Transparencia de distribución que enmascara las diferencias en la representación de datos y en los mecanismos de invocación para permitir el interfuncionamiento entre objetos.

4.4.1.2 Transparencia de fallo: Transparencia de distribución que enmascara, para que no sean percibidas por un objeto, el fallo y la posible recuperación de otros objetos (o la suya propia), a fin de permitir una tolerancia a las faltas.

4.4.1.3 Transparencia de ubicación: Transparencia de distribución que enmascara la utilización de información sobre ubicación en el espacio cuando se está identificando y vinculando a interfaces.

4.4.1.4 Transparencia de traslado: Transparencia de distribución que enmascara, para que no sea percibida por un objeto, la aptitud de un sistema para cambiar la ubicación de ese objeto. El traslado suele utilizarse para conseguir el equilibrio de la carga y reducir los efectos de errores no detectados («latentes»).

4.4.1.5 Transparencia de reubicación: Transparencia de distribución que enmascara la reubicación de una interfaz para que no sea percibida por otras interfaces vinculadas a ésta.

4.4.1.6 Transparencia de replicación: Transparencia de distribución que enmascara la utilización de un grupo de objetos mutuamente compatibles en comportamiento, para soportar una interfaz. La replicación se utiliza a menudo para mejorar el rendimiento y la disponibilidad.

4.4.1.7 Transparencia de persistencia: Transparencia de distribución que enmascara, para que no sea percibida por un objeto, la desactivación y reactivación de otros objetos (o las suyas propias). La desactivación y reactivación a menudo se utilizan para asegurar la persistencia de un objeto cuando un sistema es incapaz de proporcionarla continuamente con funciones de procesamiento, almacenamiento y comunicación.

4.4.1.8 Transparencia de transacción: Transparencia de distribución que enmascara la coordinación de actividades entre los objetos que forman una configuración, para conseguir la coherencia.

4.4.2 Utilización de la transparencia de distribución

La transparencia de distribución es un importante requisito de los usuarios de extremo en sistemas distribuidos. Este modelo de referencia define un conjunto de transparencias de distribución que hacen posible la implementación de sistemas ODP que son transparentes a la distribución desde el punto de vista de los usuarios de esos sistemas. La transparencia de distribución es selectiva; el modelo de referencia incluye reglas para seleccionar y combinar transparencias de distribución en sistemas ODP.

Este modelo de referencia contiene, para cada transparencia de distribución definida en 4.4.1.1 a 4.4.1.8, definiciones de:

- un esquema para expresar los requisitos que debe cumplir una transparencia determinada; y
- un proceso de refinamiento para transformar una especificación que describe los requisitos que debe cumplir una determinada transparencia de distribución en una especificación que realiza explícitamente el enmascaramiento implicado por esa transparencia.

NOTAS

1 En algunos casos (por ejemplo, en la transparencia de acceso), el esquema es nulo; en otros (por ejemplo, en la transparencia de transacción), el esquema contiene uno o más parámetros que establecen con precisión la forma de transparencia requerida.

2 El proceso de refinamiento comprende por lo general la introducción, en la especificación, de un comportamiento adicional, incluida la utilización de una o más funciones ODP.

Las especificaciones de los procesos de refinamiento en la cláusula 16 son prescriptivas en cuanto al nivel requerido para asegurar la integridad global del modelo de referencia.

4.5 Normas derivadas del marco

Este modelo de referencia proporciona un marco para la definición de nuevas normas y la utilización de normas existentes como normas ODP.

Las siguientes son normas ODP:

- normas para componentes de sistemas ODP;
- normas para la formación de componentes de sistemas ODP;
- normas para el modelado y la especificación de sistemas ODP.

Las normas ODP utilizan:

- el lenguaje de empresa para especificar políticas;
- el lenguaje de información para especificar la utilización e interpretación coherentes de información en las normas, y entre las normas;
- el lenguaje computacional para especificar la configuración y el comportamiento de las interfaces;
- el lenguaje de ingeniería para especificar las infraestructuras requeridas;
- el lenguaje de tecnología para especificar la conformidad con especificaciones internacionales, privadas o consensuales.

Las normas para metodología, modelado, programación, implementación y prueba de sistemas ODP utilizan el marco en su conjunto.

Las normas ODP pueden basarse en un subconjunto de este modelo de referencia (por ejemplo, excluyendo algunas formas de interacción, y determinadas funciones o transparencias). Esas normas pueden también tener un mayor alcance que el de este modelo de referencia, a condición de que las ampliaciones (o extensiones) que introduzcan no cambien ni contradigan sus disposiciones. Las ampliaciones relacionarán los nuevos términos con los términos definidos en este modelo de referencia: por ejemplo, introduciendo nuevos tipos y nuevas reglas de tipos.

Las normas ODP cumplen todos los enunciados prescriptivos de este modelo de referencia.

4.6 Conformidad

Los lenguajes de empresa, información, computacional y de ingeniería se utilizan para especificar los requisitos de conformidad para sistemas ODP. El lenguaje de tecnología puede utilizarse para afirmar la certeza de la conformidad con normas ODP en sistemas ODP. Cada interfaz que se define como un punto de conformidad tiene una especificación de información para permitir la interpretación de las interacciones de esa interfaz. Las reglas para identificar los puntos de conformidad se indican en el lenguaje computacional y en el lenguaje de ingeniería.

Un sistema ODP es conforme con una norma ODP si satisface los requisitos de conformidad de esa norma.

5 Lenguaje de empresa

El lenguaje de empresa comprende conceptos, reglas y estructuras para la especificación de un sistema ODP desde el punto de vista de la empresa.

Una especificación de empresa define la finalidad, el alcance y las políticas de un sistema ODP.

En este modelo de referencia, la prescripción desde el punto de vista de la empresa está limitada a un pequeño conjunto básico de conceptos y reglas sobre el alcance y la naturaleza de las especificaciones de empresa.

5.1 Conceptos

El lenguaje de empresa contiene los conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2 y los definidos en la presente Especificación, sujetos a las reglas de 5.2.

5.1.1 Comunidad: Configuración de objetos formada para alcanzar un objetivo. El objetivo se expresa como un contrato que especifica la manera de alcanzar el objetivo.

5.1.2 Federación <X>: Una comunidad de dominios <x>.

5.2 Reglas de estructuración

Una especificación de empresa define, y el lenguaje de empresa puede expresar, la finalidad, el alcance y las políticas de un sistema ODP, basándose en todo lo siguiente:

- los papeles (roles) desempeñados por el sistema;
- las actividades realizadas por el sistema;
- los enunciados de las políticas sobre el sistema, incluidos los relativos a los contratos de entorno.

En una especificación de empresa, un sistema ODP y el entorno en que éste opera se representan como una comunidad. En algún nivel de descripción, el sistema ODP se representa como un objeto (de) empresa en la comunidad. Los objetivos y alcance del sistema ODP en términos de los papeles que éste desempeña dentro de la comunidad de la que forma parte, y de los enunciados de política sobre estos papeles. Una comunidad se define en base a todo lo siguiente:

- los objetos de empresa que forman la comunidad;
- los papeles desempeñados por cada uno de esos objetos;
- las políticas que rigen las interacciones entre los objetos de empresa que desempeñan papeles;
- las políticas que rigen la creación, utilización y supresión de recursos por objetos de empresa que desempeñan papeles;
- las políticas que rigen la configuración de objetos de empresa y la asignación de papeles a los objetos de empresa;
- políticas relativas a contratos de entorno que rigen el sistema.

Un papel (o rol) se define en términos de los permisos, obligaciones, prohibiciones y comportamiento del objeto de empresa que desempeña el papel. Un objeto de empresa puede desempeñar uno o más papeles en una comunidad, y los papeles que ésta puede desempeñar se determinan por el contrato en que se basa la comunidad. Mientras forma parte de una comunidad, el objeto de empresa puede continuar desempeñando papeles en otras comunidades, sujeto a las disposiciones de los contratos de las comunidades participantes. El objeto de empresa puede desempeñar diferentes papeles en diferentes comunidades. Las interacciones entre objetos de empresa que desempeñan papeles apropiados dentro de diferentes comunidades pueden considerarse interacciones entre esas comunidades.

NOTA 1 – Son ejemplos de papeles (o roles) los de administrador de políticas, presidente, proveedor de servicio, propietario, gestor, accionista, consumidor.

NOTA 2 – Son ejemplos de contratos de entorno en especificaciones de empresa los requisitos de seguridad, requisitos legislativos y códigos de práctica.

NOTA 3 – En una especificación de empresa, el término «objeto<x>», donde <x> es un papel (o rol), se interpreta con el significado de «un objeto de empresa que desempeña un papel <x>»; cuando un objeto de empresa desempeña múltiples papeles, los nombres pueden concatenarse, por ejemplo, «owner driver object».

Cuando desempeña un papel, un objeto está sujeto a permisos, obligaciones y prohibiciones por delegación o transferencia. En algunos papeles, se permite que los objetos cambien las políticas. Hay cinco tipos fundamentales de acciones en materia contractual:

- un objeto asume una obligación a favor de otro (para ello tiene que permitirse concurrentemente la posibilidad de asumir la obligación);
- un objeto cumple una obligación a favor de otro objeto;
- un objeto elude una obligación a favor de otro objeto;
- un objeto recibe permiso de otro objeto para realizar alguna acción que se le tenía prohibida;
- se prohíbe a un objeto que realice una acción que anteriormente se le tenía permitida.

NOTA 4 – Un caso especial importante de permiso recibido es cuando la acción permitida es «delegativa», es decir, cuando a un objeto que desempeña un papel determinado se le permite que dé ulteriores permisos o asuma ulteriores obligaciones a nombre de un objeto que desempeña un papel superior. Esto conduce a la noción de agencia o delegación.

Las obligaciones incluyen la contabilidad y tarificación por el uso de recursos. La facturación y el pago se modelan como la reasignación de recursos entre objetos de acuerdo con los papeles que desempeñan.

Un recurso es consumible o no consumible. Un recurso consumible desaparece cuando es utilizado hasta cierto punto. En una federación <x>, el objetivo define los recursos que cada dominio <x> en la federación comparte con otros miembros de la federación. El objetivo puede dejar que cada dominio conserve un grado definido de autonomía en la utilización de sus propios recursos. El comportamiento de establecimiento para una federación <x> puede conferir autonomía a cada dominio <x> participante para que decida si ha de formar parte o no de la federación.

5.3 Conformidad y puntos de referencia

Los enunciados de conformidad en el lenguaje de empresa requieren que el comportamiento de un sistema ODP sea conforme con un determinado conjunto de objetivos y políticas.

Un implementador que pretende la conformidad debe identificar los puntos de referencia de ingeniería que dan acceso al sistema y las especificaciones de ingeniería, computacionales y de información que se aplican a dichos puntos. Por este acto, los puntos de referencia identificados pasan a ser puntos de conformidad. Las interacciones de estos puntos de conformidad pueden interpretarse entonces en términos del lenguaje empresa para comprobar que la especificación de empresa no ha sido violada.

Las especificaciones de empresa pueden aplicarse a las cuatro clases de punto de referencia (puntos de referencia programática, perceptual, de interfuncionamiento y de intercambio) definidos en la Rec. UIT-T X.902 | ISO/CEI 10746-2.

6 Lenguaje de información

El lenguaje de información comprende conceptos, reglas y estructuras para la especificación de un sistema ODP desde el punto de vista de la información.

Una especificación de información define la semántica de la información y la semántica del procesamiento de información en un sistema ODP.

En este modelo de referencia, la prescripción en el punto de vista de información está limitada a un pequeño conjunto básico de conceptos y reglas que tratan el alcance y la naturaleza de las especificaciones de información.

6.1 Conceptos

El lenguaje de información comprende los conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2 y los definidos en la presente Recomendación | Norma Internacional, sujetos a las reglas de 6.2.

6.1.1 Esquema invariante: Conjunto de predicados sobre uno o más objetos de información que tienen siempre que ser verdaderos. Los predicados constriñen los posibles estados y cambios de los objetos a que se aplican.

NOTA – Así, un esquema invariante es la especificación de los tipos de uno o más objetos de información que serán siempre satisfechos por cualquier comportamiento que los objetos pudieran tener.

6.1.2 Esquema estático: Especificación del estado de uno o más objetos de información, en un determinado instante, sujeta a las constricciones de cualesquiera esquemas invariantes.

NOTA – Así, un esquema estático es la especificación de los tipos de uno o más objetos de información en un determinado instante. Estos tipos son subtipos de los tipos especificados en el esquema invariante.

6.1.3 Esquema dinámico: Especificación de los cambios de estado admisibles de uno o más objetos de información, sujeta a las constricciones de cualesquiera esquemas invariantes.

NOTAS

1 El comportamiento en un sistema de información puede modelarse como la transición de un esquema estático a otro, es decir, como una reclasificación de instancias de un tipo a otro.

2 En el lenguaje de información, un cambio de estado que implica un conjunto de objetos puede considerarse como una interacción entre esos objetos. No todos los objetos que participan en la interacción necesitan cambiar el estado; algunos de los objetos pueden participar en modo lectura solamente.

6.2 Reglas de estructuración

Una especificación de información define la semántica de información y la semántica de procesamiento de información en un sistema ODP en términos de una configuración de objetos de información, el comportamiento de esos objetos y contratos de entorno para el sistema.

Una plantilla de objeto de información hace referencia a esquemas estáticos, invariantes y dinámicos. Las relaciones entre objetos de información pueden modelarse como parte del estado de esos objetos de información. Los objetos de información son o bien atómicos, o están representados como una composición de objetos de información componentes. El estado del objeto compuesto se representa por el estado combinado de sus objetos de información componentes. Una plantilla de objeto de información atómico representa un concepto para el cual no hay modelo en un determinado nivel de abstracción. Un objeto de información compuesto representa un concepto derivado expresado en términos de otros conceptos. Puesto que la composición de un objeto incluye la encapsulación, un objeto de información que es un componente de un objeto compuesto no puede ser un componente de otro objeto. En consecuencia, los objetos de información resultantes de la instanciación de una plantilla de objeto de información compuesto sólo existen como parte del objeto compuesto instanciado y no tienen sentido fuera de él.

Los cambios de estado admisibles especificados por un esquema dinámico pueden incluir la creación de nuevos objetos de información y la supresión de objetos de información comprendidos en el esquema dinámico. Los cambios de estado admisibles pueden estar sujetos a constricciones de ordenación y temporales.

NOTA 1 – El resultado del acceso al estado de uno o más objetos de información puede modelarse como la creación de un nuevo objeto de información.

En una especificación de información, la configuración de objetos de información y el comportamiento de esos objetos no tienen que ser adecuados para la distribución (por ejemplo, no es necesario que haya conceptos de fallo o de ubicación para interacciones de información).

NOTA 2 – Si una notación de información utiliza el concepto de interfaz, las interfaces definidas no pueden, ellas mismas, ser puntos de referencia; por tanto, no hay ningún compromiso de que las interfaces aparezcan en una implementación.

6.3 Conformidad y puntos de referencia

Los enunciados de conformidad en especificaciones de información requieren que el comportamiento de un sistema ODP sea conforme con un conjunto particular de esquemas invariantes, estáticos y dinámicos.

Un implementador que pretende la conformidad debe indicar los puntos de referencia de ingeniería apropiados que dan acceso al sistema y las especificaciones de ingeniería y computacionales aplicables a dichos puntos. Por este acto los puntos de referencia identificados se convierten en puntos de conformidad. Las interacciones en estos puntos de conformidad pueden interpretarse entonces en términos del lenguaje de información para comprobar que son consecuentes con los esquemas invariantes, estáticos y dinámicos.

Las especificaciones de información pueden aplicarse a las cuatro clases de puntos de referencia (puntos de referencia programática, perceptual, de interfuncionamiento y de intercambio) identificadas en la Rec. UIT-T X.902 | ISO/CEI 10746-2.

7 Lenguaje computacional

El lenguaje computacional comprende conceptos, reglas y estructuras para la especificación de un sistema ODP desde el punto de vista computacional.

Una especificación computacional define la descomposición funcional de un sistema ODP en objetos que interactúan en interfaces.

En el punto de vista computacional, las aplicaciones y funciones ODP consisten en configuraciones de objetos computacionales interactuantes.

7.1 Conceptos

El lenguaje de computación contiene los conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2 y los definidos en la presente Recomendación | Norma Internacional, sujetos a las reglas de estructuración de 7.2.

7.1.1 Señal: Acción atómica compartida que produce una comunicación en un solo sentido desde un objeto iniciador a un objeto respondedor.

NOTA – Una señal es una interacción.

7.1.2 Operación: Interacción entre un objeto (de) cliente y un objeto (de) servidor, la cual es o bien una interrogación o un anuncio.

7.1.3 Anuncio: Interacción – la **invocación** – que es iniciada por un objeto cliente y produce como resultado el transporte de información desde el objeto cliente al objeto servidor, y por la que se pide la realización de una función por ese objeto servidor.

7.1.4 Interrogación: Una interacción constituida por:

- una interacción – la **invocación** – iniciada por un objeto cliente, que produce como resultado el transporte de información desde ese objeto cliente a un objeto servidor, por la que se pide una función que será realizada por el objeto servidor, seguida de
- una segunda interacción – la **terminación** – iniciada por el objeto servidor, como resultado de la cual se produce el transporte de información desde el objeto servidor al objeto cliente en respuesta a la invocación.

NOTA – En interrogaciones, las invocaciones y terminaciones siempre aparecen por pares. Los anuncios no tienen terminaciones. En consecuencia, no es posible la existencia de una operación que esté constituida por una invocación seguida de una secuencia de terminaciones asociadas.

7.1.5 Flujo: Abstracción de una secuencia de interacciones, como resultado de la cual se produce el transporte de información desde un objeto productor a un objeto consumidor.

NOTA – Un flujo puede utilizarse para simplificar (dícese, efectuar una abstracción de) la estructura exacta de una secuencia de interacciones, o de una interacción continua que incluya el caso especial de un flujo de información analógica.

7.1.6 Interfaz de señales: Interfaz en la que todas las interacciones son señales.

7.1.7 Interfaz de operaciones: Interfaz en la que todas las interacciones son operaciones.

7.1.8 Interfaz de trenes: Interfaz en la que todas las interacciones son flujos.

7.1.9 Plantilla de objeto computacional: Plantilla de objeto que está constituida por un conjunto de plantillas de interfaz computacional que el objeto puede instanciar, una especificación de comportamiento y una especificación de contrato de entorno.

7.1.10 Plantilla de interfaz computacional: Plantilla de interfaz para una interfaz de señales, o para una interfaz de trenes, o para una interfaz de operaciones. Una plantilla de interfaz computacional está constituida por una firma interfaz de señales, de trenes de operaciones, según proceda, una especificación de comportamiento y una especificación de contrato de entorno.

7.1.11 Firma de interfaz de señales: Firma de interfaz para una interfaz de señales. Una firma de interfaz de señales comprende un conjunto finito de plantillas de acción, una para cada tipo de señal en la interfaz. Cada plantilla de acción está constituida por el nombre de la señal, el número, los nombres y tipos de sus parámetros y una indicación de causalidad (iniciador o respondedor, pero no ambos) con respecto al objeto que instancia la plantilla.

7.1.12 Firma de interfaz de operaciones: Firma de interfaz para una interfaz de operaciones. Una firma de interfaz de operaciones comprende un conjunto de firmas de anuncio y de interrogación, según proceda, una para cada tipo de operación en la interfaz, junto con una indicación de causalidad (cliente o servidor, pero no ambos) para la interfaz en su conjunto, con respecto al objeto que instancia la plantilla.

Cada **firma de anuncio** es una plantilla de acción que contiene el nombre de la invocación y el número, los nombres y los tipos de sus parámetros.

Cada **firma de interrogación** comprende una plantilla de acción con los siguientes elementos:

- el nombre de la invocación;
- el número, los nombres y los tipos de sus parámetros;
- un conjunto finito, no vacío, de plantillas de acción, una para cada tipo de terminación posible de la invocación, cada una de las cuales contiene el nombre de la terminación y el número, los nombres y los tipos de sus parámetros.

7.1.13 Firma de interfaz de trenes: Firma de interfaz para una interfaz de trenes. Una interfaz de trenes comprende un conjunto finito de plantillas de acción, una para cada tipo de flujo en el interfaz de trenes. Cada plantilla de acción para un flujo contiene el nombre del flujo, el tipo de información del flujo, y una indicación de causalidad para el flujo (es decir, productor o consumidor, pero no ambos) con respecto al objeto que instancia la plantilla.

NOTAS

1 La frase «firma de interfaz complementaria de *X*», donde *X* es en sí una firma de interfaz, describe una firma de interfaz idéntica a *X* en todos los aspectos, salvo el de la causalidad, que es opuesta a la de *X*.

2 Muchos lenguajes de definición de interfaz (IDL, *interface definition language*) sólo captan las plantillas de acción de una firma y dependen del contexto en que se utiliza el IDL para determinar la causalidad que ha de aplicarse.

7.1.14 Objeto de vinculación: Objeto computacional que soporta una vinculación entre un conjunto de otros objetos computacionales.

NOTA – Los objetos de vinculación deben cumplir disposiciones especiales (véase 7.2.3).

7.2 Reglas de estructuración

Una especificación computacional describe la descomposición funcional de un sistema ODP, en términos transparentes a la distribución, como:

- una configuración de objetos computacionales (incluidos objetos de vinculación);
- las acciones internas de esos objetos;
- las interacciones que se producen entre esos objetos;
- los contratos de entorno para esos objetos y sus interfaces.

Una especificación computacional está constreñida por las reglas del lenguaje computacional. Éstas comprenden:

- reglas de interacción (véase 7.2.2), reglas de vinculación (véase 7.2.3) y reglas de tipos (véase 7.2.4) que proporcionan un interfuncionamiento transparente a la distribución;
- reglas de plantillas (véase 7.2.5) que se aplican a todos los objetos computacionales;
- reglas de fallos (véase 7.2.6) que se aplican a todos los objetos computacionales e identifican los puntos potenciales de fallos en actividades computacionales.

Se proporcionan reglas de portabilidad (véase 7.2.7) para dar directrices a quienes elaboran normas sobre la portabilidad en ODP.

Una especificación computacional define un conjunto inicial de objetos computacionales y su comportamiento. La configuración cambia cuando los objetos computacionales:

- instancian ulteriores objetos computacionales;
- instancian ulteriores interfaces computacionales;
- realizan acciones de vinculación;
- efectúan funciones sobre objetos de vinculación;
- suprimen interfaces computacionales;
- suprimen objetos computacionales.

7.2.1 Reglas de denominación

Cada clase de nombre definido en el lenguaje informático tiene un contexto asociado, de la manera siguiente:

- un nombre de señal en una firma de interfaz de señales es un identificador en el contexto de esa firma;
- un nombre de flujo en una firma de interfaz de trenes es un identificador en el contexto de esa firma;
- un nombre de invocación en una firma de interfaz de operaciones es un identificador en el contexto de esa firma;
- un nombre de terminación en una firma de interfaz de operaciones es un identificador en el contexto de la plantilla de operación en que aparece;
- el nombre de un parámetro en una plantilla de señales es un identificador en el contexto de esa plantilla;
- el nombre de un parámetro en una plantilla de invocación en una firma de interfaz de operaciones es un identificador en el contexto de esa plantilla;
- el nombre de un parámetro en una plantilla de terminación en una firma de interfaz de operaciones es un identificador en el contexto de esa plantilla;
- el nombre de un parámetro en una plantilla de señales en una firma de interfaz de señales es un identificador en el contexto de esa plantilla.

NOTA 1 – Los nombres de señales son distintos en cualquier firma de interfaz de señales, pero las señales en firmas diferentes pueden tener el mismo nombre, y así sucesivamente.

Un identificador de interfaz computacional es inequívoco dentro de su contexto (es decir, no puede estar asociado con más de una interfaz computacional en ese contexto). La elección de contextos para identificadores de interfaz computacional es una materia propia del diseño del lenguaje, por lo que está fuera del ámbito de este modelo de referencia. En consecuencia, el modelo de referencia no impone constricciones sobre la extensión de los contextos para los identificadores de interfaz computacional. Por esta razón, no se puede confiar en:

- la extensión de contextos de denominación para identificadores de interfaz computacional (por ejemplo, suponer que estén relacionados con estructuras de lenguaje de ingeniería tales como nodos o dominios de comunicaciones);
- la unicidad de los identificadores de interfaz computacional (es decir, se permiten sinónimos);
- que un identificador de interfaz computacional indique la misma interfaz computacional en todos los lugares en que aparece el identificador (es decir, los nombres no tienen que ser «globales»).

NOTA 2 – Una determinada notación computacional puede no tener términos explícitos que denoten identificadores computacionales; por esta razón, en esa notación, los identificadores de interfaz computacional son explícitos; sin embargo, siguen estando sujetos a las reglas antes indicadas.

7.2.2 Reglas de interacción

Cada interacción de un objeto computacional aparece en una de sus interfaces computacionales. El lenguaje computacional impone constricciones al comportamiento permitido en una interfaz computacional. La interacción en una interfaz no vinculada fracasa. Las reglas de vinculación (véase 7.2.3) imponen constricciones a la manera de vincular los interfaces.

La parte del lenguaje computacional relativa a la interacción soporta tres modelos de interacciones, cada uno de los cuales tiene asociado una clase de interfaz computacional:

- señales e interfaces de señales;
- flujos e interfaces de trenes;
- operaciones e interfaces de operaciones.

Aparte de las diferentes clases de interacción soportadas, los modelos de interacción difieren en sus propiedades relativas a los fallos. Los participantes en un flujo u operación pueden tener una visión incoherente de una interacción en tiempos diferentes, especialmente cuando se han producido fallos. En contraste con los trenes y las operaciones, no hay un concepto de fallo parcial de una señal; una señal o bien tiene éxito, o bien fracasa de la misma manera para ambos participantes en la interacción.

7.2.2.1 Reglas de interacción de señales

Un objeto computacional que ofrece una interfaz de señales de un determinado tipo de interfaz de señales:

- inicia señales que tienen la causalidad de iniciador en la firma de la interfaz;
- responde a señales que tienen la causalidad de respondedor en la firma de la interfaz.

7.2.2.2 Reglas de interacción de trenes

Un objeto computacional que ofrece una interfaz de trenes:

- genera flujos que tienen la causalidad de productor en la firma de la interfaz;
- recibe flujos que tienen la causalidad de consumidor en la firma de la interfaz.

7.2.2.3 Reglas de interacción de operaciones

Un objeto cliente que utiliza una interfaz de operaciones invoca las operaciones denominadas en la firma de la interfaz. Un objeto servidor que ofrece una interfaz de operaciones espera cualquiera de las operaciones denominadas en la firma de la interfaz. En el caso de una interrogación, el servidor responde a la invocación iniciando cualquiera de las terminaciones denominadas para la operación en la firma de la interfaz del servidor. El cliente espera cualquiera de las terminaciones denominadas para la operación en la firma de la interfaz del cliente. La duración de la operación es arbitraria a menos que los contratos de entorno aplicables a los objetos y a las interfaces participantes requieran otra cosa.

NOTA – Si un hilo de cliente («client thread») invoca una cadena de interrogación, la «toma de contacto» bidireccional de la invocación y la terminación asegura que las operaciones serán respondidas por el servidor en el mismo orden en que fueron iniciadas por el cliente. Si el cliente invoca una cadena de anuncios (o una cadena que contiene anuncios e invocaciones) no hay toma de contacto para garantizar el orden en que el servidor responderá a los anuncios, a menos que esto esté implícito en los contratos de entorno aplicables a la interacción. No se garantiza que las operaciones serán respondidas en el mismo orden, tanto en el caso de interrogaciones como en el de anuncios, cuando formen parte de actividades que desciendan por ramas diferentes de una anterior acción de división.

7.2.2.4 Reglas de parámetros

Los parámetros para señales, invocaciones y terminaciones pueden incluir identificadores para interfaces computacionales y tipos de firma de interfaz computacional.

NOTA 1 – La posibilidad de parámetros de firma de interfaz computacional hace que los sistemas de tipo de firma computacional sean de un orden más alto. Una representación explícita de tipos de firma se requiere, por ejemplo, en comercio («trading»), donde los parámetros de operaciones de importación y exportación incluyen tipos de firma:

```
trader.import (T: Type, ...) : (service: T) -> failed (reason: String)
trader.export (T: Type, service: T) : (...) -> failed (reason: String)
```

Esto crea la necesidad de una comprobación dinámica del subtipo de firma (véase 7.2.5.1).

Un parámetro formal que es un identificador para una interfaz computacional es calificado por un tipo de firma de interfaz computacional. El correspondiente parámetro real tiene que referenciar una interfaz con un tipo de firma de interfaz (o uno de sus subtipos). El parámetro real sólo puede utilizarse como si hiciera referencia a una interfaz computacional con el mismo tipo de firma que el parámetro formal (o uno de los supertipos del parámetro formal). Después de una interacción, tanto el iniciador como el respondedor pueden referenciar la interfaz identificada, aunque posiblemente lo hagan por diferentes identificadores de interfaz informática.

NOTA 2 – Esta regla impide que el usuario de la interfaz referenciada por el parámetro real esté en condiciones para realizar interacciones adicionales más allá de las correspondientes al tipo de firma de interfaz formal, aunque la interfaz referenciada por el parámetro real es un subtipo del tipo de firma de interfaz asociado con el parámetro formal.

7.2.2.5 Flujos, operaciones y señales

Se pueden definir flujos y operaciones en términos de señales. Esto permite utilizar interfaces de señales como base para explicar la vinculación multipartita, las características de calidad de servicio de extremo a extremo y las vinculaciones compuestas entre diferentes clases de interfaz (por ejemplo vinculaciones de interfaz de trenes a interfaz de operaciones).

La definición de los flujos que utilizan señales depende de los detalles de las interacciones formuladas por simplificación (o abstracción) en la especificación de la interfaz de trenes en cuestión y, por tanto, está fuera del ámbito de este modelo de referencia.

Para modelar operaciones por medio de señales se introducen las correspondientes interfaces de señales en la interfaz de operaciones del cliente y en la interfaz de operaciones del servidor participantes:

- en una interfaz de señales correspondiente a una interfaz de operaciones de cliente hay una señal (**depósito de invocación**) que corresponde a cada invocación con los mismos parámetros y, en el caso de una interfaz que contiene interrogaciones, una señal (**entrega de terminación**) que corresponde a cada posible terminación con los mismos parámetros que esa terminación;
- en la interfaz de señales correspondiente a una interfaz de operaciones de servidor hay una señal (**entrega de invocación**) que corresponde a cada invocación con los mismos parámetros y, en el caso de una interfaz que contiene interrogaciones, una señal (**depósito de terminación**) que corresponde a cada posible terminación con los mismos parámetros que esa terminación.

Esto crea una equivalencia entre el conjunto resultante de señales y el conjunto de invocaciones y terminaciones en las interfaces de operaciones que se describen.

7.2.3 Reglas de vinculación

En este modelo de referencia, la vinculación se define con referencia a acciones de vinculación. La utilización de estas acciones se denomina **vinculación explícita**. Hay dos clases de acciones de vinculación: las **acciones de vinculación primitiva** y las **acciones de vinculación compuesta**.

Una acción de vinculación primitiva vincula directamente dos objetos computacionales. Una acción de vinculación compuesta puede expresarse en términos de acciones de vinculación primitivas que enlazan dos o más objetos computacionales a través de un objeto de vinculación. La presencia de un objeto de vinculación en una vinculación computacional da el medio para expresar el control de la configuración y de la calidad de servicio (véase 7.2.3.3).

En las notaciones en que no hay términos para expresar acciones de vinculación, la vinculación es **implícita**. La **vinculación implícita** para otras interfaces diferentes de las interfaces de operaciones de servidor no está definida en este modelo de referencia, porque en otros casos no es evidente en donde habrá de situarse la iniciativa, en la vinculación, con relación a la interacción subsiguiente. La información adicional necesaria puede suministrarse en una acción de vinculación explícita.

7.2.3.1 Reglas de vinculación implícita para interfaces de operaciones de servidor

Si una invocación por un objeto cliente referencia una interfaz de operaciones de servidor con la que el cliente no está vinculado, se requiere una vinculación implícita. El establecimiento de una vinculación implícita comprende el siguiente procedimiento si no existe una interfaz adecuada de operaciones de cliente vinculada al servidor:

- se crea una interfaz de operaciones de cliente de tipo de firma complementaria a la de la interfaz de servidor;
- se vincula la interfaz de operaciones de cliente a la interfaz de operaciones de servidor;
- se invoca el objeto servidor utilizando la interfaz de operaciones de cliente;
- (facultativamente) una vez concluida la operación, se suprime la interfaz de cliente.

7.2.3.2 Reglas de vinculación primitiva

Las acciones de vinculación primitiva hacen posible la vinculación de una interfaz del objeto que realiza la acción a otra interfaz (de otro objeto, o de ese mismo objeto). La acción de vinculación es parametrizada por dos identificadores, uno para cada interfaz que interviene. Las precondiciones para una acción de vinculación primitiva son que las dos interfaces participantes sean de la misma clase (esto es, de señales, de trenes o de operaciones), que sean de causalidad complementaria y que sus tipos de firma sean complementarios.

Una vinculación primitiva o bien establece una vinculación entre las dos interfaces que intervienen, o fracasa.

La supresión de una interfaz que ha sido vinculada a otra interfaz utilizando una acción de vinculación primitiva suprime la vinculación junto con la interfaz.

7.2.3.3 Reglas para la vinculación compuesta

Las acciones de vinculación compuesta permiten vincular un conjunto de interfaces utilizando un objeto de vinculación para soportar la vinculación. Salvo las disposiciones de esta cláusula, un objeto de vinculación es un objeto computacional ordinario. En una plantilla de objeto de vinculación, la especificación de comportamiento se expresa en términos de un conjunto de parámetros de roles formales, cada uno de los cuales está asociado con una plantilla de interfaz.

Las acciones de vinculación compuesta son parametrizadas por una plantilla de objeto de vinculación y un conjunto de interfaces que serán vinculadas para interacción.

Las precondiciones para la vinculación compuesta son las siguientes, para cada rol formal en la plantilla de objeto de vinculación:

- el parámetro de interfaz correspondiente tiene que ser de la misma clase (es decir, de señales, de trenes o de operaciones) que el de la plantilla asociada con el rol formal en la plantilla de objeto de vinculación;
- el parámetro de interfaz correspondiente tiene que ser de causalidad complementaria a la plantilla de interfaz asociada con el rol formal en la plantilla de objeto de vinculación;
- el parámetro de interfaz correspondiente tiene que ser un subtipo del tipo de firma de la plantilla de interfaz asociada con el rol formal en la plantilla de objeto de vinculación.

Una acción de vinculación compuesta comprende los siguientes pasos:

- se instancia un objeto de vinculación a partir de la plantilla de objeto de vinculación;
- se instancia cada plantilla de interfaz dentro del objeto de vinculación asociado con un parámetro de rol formal en la plantilla de objeto de vinculación;
- el objeto de vinculación utiliza acciones de vinculación primitiva para vincular cada una de estas interfaces a la interfaz referenciada en el parámetro real correspondiente;
- se instancia un conjunto de interfaces de control y los identificadores para estas interfaces se retornan como resultado de la acción de vinculación (es decir, pasan a formar parte del estado del objeto que realizó la acción; este objeto puede subsiguientemente pasar el identificador por interacción con otros objetos computacionales).

Las interfaces de control de un objeto de vinculación proporcionan algunas de las funciones siguientes, o todas ellas:

- supervisión de la utilización de la vinculación;
- supervisión de las modificaciones de la vinculación;
- autorización de modificaciones de la vinculación;
- cambio de los participantes en la vinculación;
- cambio del patrón de comunicación habilitado por la vinculación;
- cambio de la calidad de servicio de la vinculación;
- supresión de la vinculación en su totalidad.

El efecto de la supresión de una vinculación a un objeto de vinculación lo determina el comportamiento del objeto de vinculación.

7.2.4 Reglas de tipos

Este modelo de referencia especifica reglas relativas a los tipos de firma para interfaces computacionales. Las reglas para el establecimiento de subtipos de firmas (brevemente, reglas de subtipificación de firmas) definen requisitos mínimos para que una interfaz pueda remplazar a otro. Estas reglas se basan en la semántica de interacción de las interfaces computacionales (es decir, interfaces de señales, de trenes y de operaciones). Son suficientes para asegurar que la interfaz reemplazante pueda interpretar coherentemente la estructura de cualquier interacción que se produzca.

Se pueden definir reglas para la subtipificación de firmas para interfaces con semánticas de interacción alternativas en términos de señales; estas definiciones pueden introducirse mediante una norma ODP.

7.2.4.1 Reglas de subtipificación de firmas de interfaz de señales

La definición de subtipos de firmas de interfaz de señales se presenta en el Anexo A. Seguidamente se resumen las reglas para los tipos de interfaz de señales que no se hayan definido recursivamente.

Una firma de interfaz de señales de tipo X es un subtipo de una firma de interfaz de señales de tipo Y si se cumplen las siguientes condiciones:

- para cada firma de señal iniciadora en Y hay una firma de señal iniciadora correspondiente en X que tiene el mismo nombre, y el mismo número y nombres de parámetros, y se cumple que cada tipo de parámetro en X es un subtipo del tipo de parámetro correspondiente en Y ;
- para cada firma de señal respondedora en X hay una firma de señal respondedora correspondiente en Y que tiene el mismo nombre, y el mismo número y nombres de parámetros, y se cumple que cada tipo de parámetro en Y es un subtipo del tipo de parámetro correspondiente en X .

7.2.4.2 Reglas de subtipificación de firmas de interfaz de trenes

Las reglas de subtipificación de firmas de interfaz de trenes dependen de los detalles de las interacciones formuladas abstractamente en la definición de las interfaces de trenes que intervienen. En particular, estos detalles aclararán si las reglas de subtipificación permitirán o no correspondencias incompletas entre el conjunto de flujos en las dos interfaces. Por consiguiente, unas reglas completas de subtipificación de firmas están fuera del ámbito de este modelo de referencia. Las constricciones relativas a la subtipificación de firmas de trenes se indican en el Anexo A. Seguidamente se resumen las constricciones para los tipos de interfaz de trenes que no se hayan definido recursivamente.

La interfaz de trenes *X* es un subtipo de firma de la interfaz de trenes *Y* si se cumplen las condiciones siguientes para todos los flujos que tienen nombres idénticos:

- si la causalidad es productor, el tipo de información en *X* es un subtipo del tipo de información en *Y*;
- si la causalidad es consumidor, el tipo de información en *Y* es un subtipo del tipo de información en *X*.

7.2.4.3 Reglas de subtipificación de firmas de interfaz de operaciones

La definición de la subtipificación de firmas de interfaz de operaciones se da en el Anexo A. Seguidamente se resumen las reglas para los tipos de interfaz de operaciones que no se hayan definido recursivamente.

Una interfaz de operaciones *X* es un subtipo de firma de la interfaz *Y* si se cumplen las siguientes condiciones:

- para cada interrogación en *Y*, hay una firma de interrogación en *X* (la firma correspondiente en *X*) que define una interrogación con el mismo nombre;
- para cada firma de interrogación en *Y*, la firma de interrogación correspondiente en *X* tiene el mismo número y nombres de parámetros;
- para cada firma de interrogación en *Y*, cada tipo de parámetro es un subtipo del correspondiente tipo de parámetro en la firma de interrogación correspondiente en *X*;
- el conjunto de nombres de terminación de una firma de interrogación en *Y* contiene el conjunto de nombres de terminación de la correspondiente firma de interrogación en *X*;
- para cada firma de interrogación en *Y*, una terminación dada en la correspondiente firma de interrogación en *X* tiene el mismo número y nombres de parámetros de resultado en la terminación del mismo nombre en la firma de interrogación en *Y*;
- para cada firma de interrogación en *Y*, cada tipo de resultado asociado con una terminación dada en la correspondiente firma de interrogación en *X* es un subtipo del tipo de resultado (con el mismo nombre) en la terminación con el mismo nombre en *Y*;
- para cada anuncio en *Y*, hay una firma de anuncio en *X* (la firma correspondiente en *X*) que define un anuncio con el mismo nombre;
- para cada firma de anuncio en *Y*, la correspondiente firma de anuncio en *X* tiene el mismo número y nombres de parámetros;
- para cada firma de anuncio en *Y*, cada tipo de parámetro es un subtipo del correspondiente tipo de parámetro en la correspondiente firma de anuncio en *X*.

7.2.5 Reglas de plantillas

7.2.5.1 Reglas de plantillas de objeto computacional

Un objeto computacional (incluido el caso especial de un objeto de vinculación) puede:

- iniciar señales o responder a señales;
- producir o consumir flujos;
- iniciar invocaciones de operación;
- responder a invocaciones de operación;
- iniciar terminaciones de operación;
- responder a terminaciones de operación;
- instanciar plantillas de interfaz;
- instanciar plantillas de objeto;
- vincular interfaces;

- ganar acceso a su estado y modificarlo;
- suprimir una o más de sus interfaces;
- suprimirse él mismo;
- realizar ramificaciones permanentes, ramificaciones temporales y junción de actividades;
- obtener un identificador de interfaz computacional para una instancia de la función de comercio;
- probar si una firma de interfaz computacional es un subtipo de otra.

Cualquiera de estas acciones puede fracasar.

7.2.5.2 Instanciación de interfaz computacional

Una instanciación de una interfaz computacional establece uno o más identificadores de interfaz computacional para la nueva interfaz en el objeto que efectúa la instanciación.

7.2.5.3 Instanciación de plantilla de objeto computacional

La expresión de comportamiento en una plantilla de objeto computacional incluye una descripción del comportamiento que se produce cuando se instancia la plantilla (el **comportamiento instanciado**). La especificación del contrato de entorno describe el contrato que habrá de establecerse entre el objeto instanciado y su entorno cuando se instancia la plantilla. Cuando el comportamiento de instanciación incluye instanciación de interfaz, la instanciación establece identificadores para estas interfaces en el objeto que inició la instanciación.

7.2.6 Reglas de fallos

Los modos de fallo visibles para un objeto vienen determinados por sus especificaciones de comportamiento y de contrato de entorno.

Cualesquiera de las acciones computacionales descritas en 7.2.5.1 pueden fracasar y ese fallo puede ser observado por el objeto que realiza la acción. Una interacción puede ser interrumpida por el fallo de los objetos que en ella intervienen, o de la vinculación entre ellos, o por ambas causas. En el caso de señales, el fallo es idéntico y visible para todos los participantes en la interacción. En el caso de flujos y operaciones, el fallo no se produce necesariamente en todos los participantes, y puede aparecer en tiempos diferentes y con parámetros diferentes para cada participante que sufra el fallo.

NOTA – Son ejemplos de fallos de interacción los fallos de seguridad, los fallos de comunicación y los fallos de recursos.

En el caso de operaciones, el fallo del objeto computacional servidor al no responder a invocaciones o no iniciar terminaciones puede ser observado por el objeto computacional cliente participante.

La instanciación de una plantilla de objeto o de una plantilla de interfaz computacional fracasa si no se puede satisfacer el contrato de entorno. Una acción de vinculación fracasa si no se puede satisfacer cualquiera de sus contratos de entorno en las interfaces que están siendo vinculadas.

7.2.7 Reglas de portabilidad

Las normas de portabilidad en sistemas ODP especifican plantillas de acción para las acciones descritas en 7.2.5.1. La especificación de esas plantillas es una materia propia del diseño de lenguajes y, por tanto, está fuera del ámbito de este modelo de referencia. Además de las cuestiones relativas a la sintaxis, una norma sobre portabilidad debe tratar cuestiones semánticas específicas, como son:

- las reglas de composición para plantillas de acción, incluidas las plantillas para acciones de ramificación temporal y de junción con el fin de hacer posible la concurrencia y la sincronización;
- los términos disponibles para la especificación de plantillas de objeto y de interfaz, junto con las reglas para su composición;
- garantías para la ordenación y la entrega de los anuncios.

Una norma de portabilidad puede representar las acciones permitidas directamente (por ejemplo, como funciones de biblioteca) o indirectamente mediante estructuras sintácticas. Puede haber otras normas de portabilidad establecidas en términos de estilo (por ejemplo, un modelo de procesamiento basado en eventos por oposición a un modelo basado en hilos) y también en términos del contenido (por ejemplo, el número de acciones computacionales soportadas). Este modelo de referencia identifica dos normas de portabilidad.

Una **norma de portabilidad básica** es una norma de portabilidad que incluye por lo menos:

- interrogaciones;
- vinculación implícita;

- instanciación de objetos computacionales;
- instanciación de interfaces computacionales;
- acceso al estado y modificación del estado;
- soporte de hilos mediante acciones de ramificación permanente, ramificación temporal y junción;
- obtención de un identificador para una interfaz computacional en que se proporcionan funciones de comercio (para permitir una subsiguiente vinculación a la función y su utilización);
- prueba del tipo de firma de interfaz.

Una **norma de portabilidad extendida** es una norma de portabilidad que incluye todas las acciones descritas en 7.2.5.1.

7.3 Conformidad y puntos de referencia

En el lenguaje computacional hay un punto de referencia en todo interfaz de todo objeto. Cada punto de referencia puede tornarse en un punto de conformidad programática, un punto de conformidad perceptual, un punto de conformidad de interfuncionamiento o un punto de conformidad de intercambio, lo que depende de los requisitos establecidos cuando se designa el punto de referencia como punto de conformidad por una norma determinada o en la especificación del sistema.

En el lenguaje computacional, estos requisitos se especifican en términos de las plantillas de interfaz y de objeto que determinan la interfaz del objeto conforme.

Un implementador que alega la conformidad con una especificación computacional debe indicar los puntos de referencia que corresponden al punto de referencia computacional requerido y expresar las estructuras de transparencia y de ingeniería que se aplican a esos puntos. Por este acto, los puntos de referencia identificados se convierten en puntos de conformidad. El conjunto de interacciones en estos puntos de conformidad puede interpretarse entonces en términos del lenguaje informático para determinar que la especificación informática no ha sido violada.

La conformidad de un objeto en un punto de conformidad programática puede probarse en términos de un lenguaje de especificación de interfaz normalizado y de una vinculación de lenguaje que satisfaga las reglas de portabilidad. La conformidad de un objeto en un punto de conformidad de interfuncionamiento puede probarse en términos de interacciones visibles en protocolos de comunicaciones.

8 Lenguaje de ingeniería

El lenguaje de ingeniería comprende conceptos, reglas y estructuras para la especificación de un sistema ODP desde el punto de vista de la ingeniería.

Una especificación de ingeniería define los mecanismos y funciones requeridos para soportar la interacción distribuida entre objetos en un sistema ODP.

8.1 Conceptos

El lenguaje de ingeniería contiene los conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2 y los definidos en la presente Recomendación | Norma Internacional, sujetos a las reglas de 8.2.

8.1.1 Objeto de ingeniería básico: Objeto de ingeniería que requiere el soporte de una infraestructura distribuida.

8.1.2 Conglomerado («cluster»): Configuración de objetos de ingeniería básicos que forman una sola unidad para fines de desactivación, verificación por punto de comprobación, reactivación, recuperación y traslado.

NOTA – Un ejemplo de conglomerado es un segmento de memoria virtual que contiene objetos.

8.1.3 Gestor de conglomerado: Objeto de ingeniería que gestiona los objetos de ingeniería básicos en un conglomerado.

8.1.4 Cápsula: Configuración de objetos de ingeniería que forman una sola unidad para fines de encapsulación de procesamiento y almacenamiento.

NOTA – Un ejemplo de cápsula es una máquina virtual (por ejemplo, un proceso).

8.1.5 Gestor de cápsula: Objeto de ingeniería que gestiona los objetos de ingeniería en una cápsula.

8.1.6 Núcleo: Objeto de ingeniería que coordina funciones de procesamiento, almacenamiento y comunicaciones para uso por otros objetos de ingeniería dentro del nodo a que pertenece.

NOTA – Un ejemplo de núcleo es un sistema operativo («kernel»).

8.1.7 Nodo: Configuración de objetos de ingeniería que forman una sola unidad a los fines de ubicación en el espacio, y que incorpora un conjunto de funciones de procesamiento, almacenamiento y comunicación.

NOTAS

- 1 Un ejemplo de nodo es un computador y el software que soporta (sistema operativo y aplicaciones).
- 2 Un nodo puede tener una estructura interna que no interese a una especificación de ingeniería. Así, por ejemplo, un nodo puede ser un computador paralelo bajo el control de un solo sistema operativo.

8.1.8 Canal: Configuración de stubs, vinculadores, objetos de protocolo e interceptores que proporcionan una vinculación entre un conjunto de interfaces y objetos de ingeniería básicos, a través de los cuales puede producirse la interacción.

NOTA – Las vinculaciones que requieren canales se denominan **vinculaciones distribuidas** en el lenguaje de ingeniería; las vinculaciones entre objetos de ingeniería que no requieren canales (por ejemplo, entre objetos de ingeniería en el mismo conglomerado) se denominan **vinculaciones locales**.

8.1.9 Stub: Objeto de ingeniería en un canal, que interpreta las interacciones transportadas por dicho canal, y, basándose en esta interpretación, realiza cualquier transformación o supervisión necesaria.

NOTA – Por ejemplo, un stub puede organizar parámetros introduciéndolos en memorias tampón de comunicaciones, o registrar actividades para fines de auditoría.

8.1.10 Vinculador («binder»): Objeto de ingeniería en un canal, que mantiene una vinculación distribuida entre objetos de ingeniería básicos interactuantes.

8.1.11 Interceptor <x>: Objeto de ingeniería en un canal, situado en una frontera entre dominios <x>. Un interceptor <x> efectúa:

- comprobaciones para hacer cumplir o supervisar políticas sobre interacciones permitidas entre objetos de ingeniería en dominios diferentes;
- transformaciones para enmascarar diferencias en la interpretación de datos por objetos de ingeniería básicos en dominios diferentes.

NOTA – Ejemplo de un interceptor es un relevador entre subredes.

8.1.12 Objeto de protocolo: Objeto de ingeniería en un canal, que comunica con otros objetos de protocolo en el mismo canal para lograr la interacción entre objetos de ingeniería básicos (que pudieran estar en conglomerados diferentes, o en cápsulas diferentes, o en nodos diferentes).

8.1.13 Dominio de comunicaciones: Conjunto de objetos de protocolo capaces de interfuncionar.

8.1.14 Interfaz de comunicación: Interfaz de un objeto de protocolo que puede estar vinculada a una interfaz de un objeto interceptor o de otro objeto de protocolo en un punto de referencia de interfuncionamiento.

8.1.15 Identificador de punto extremo de vinculación: Identificador, en el contexto de denominación de una cápsula, que es utilizado por un objeto de ingeniería básico para seleccionar una de las vinculaciones en que participa, para fines de interacción.

NOTAS

- 1 Un ejemplo de identificador de punto extremo de vinculación es una dirección de memoria (para una estructura de datos que representa una interfaz de ingeniería).
- 2 Puede utilizarse la misma forma de identificador de punto extremo de vinculación independientemente de que la vinculación en que se participa sea local o distribuida.

8.1.16 Referencia de interfaz de ingeniería: Identificador, en el contexto de un dominio de gestión de referencias de interfaz de ingeniería, para una interfaz de objeto de ingeniería que está disponible para vinculación distribuida.

NOTA – Una referencia de interfaz de ingeniería se necesita para establecer vinculaciones distribuidas, y es distinta de los identificadores de punto de extremo de vinculación utilizados por un objeto de ingeniería básico para fines de interacción.

8.1.17 Dominio de gestión de referencias de interfaz de ingeniería: Conjunto de nodos que forman un dominio de denominación para fines de asignación de referencias de interfaz de ingeniería.

8.1.18 Política de gestión de referencias de interfaz de ingeniería: Conjunto de permisos y prohibiciones que rigen la federación de dominios de gestión de referencias de interfaz de ingeniería.

8.1.19 Plantilla de conglomerado («cluster template»): Plantilla de objeto para una configuración de objetos y toda actividad requerida para instanciar esos objetos y establecer vinculaciones iniciales.

8.1.20 Punto de comprobación («checkpoint»): Plantilla de objeto derivada del estado y de la estructura de un objeto de ingeniería y que puede utilizarse para instanciar otro objeto de ingeniería, coherente con el estado del objeto original en el momento de la verificación por punto de comprobación.

8.1.21 Verificación por puntos de comprobación («checkpointing»): Creación de un punto de comprobación. Sólo pueden crearse puntos de comprobación cuando el objeto de ingeniería que interviene satisface una precondición enunciada en una política de verificación por puntos de comprobación.

8.1.22 Punto de comprobación de conglomerado: Plantilla de conglomerado que contiene puntos de comprobación de los objetos de ingeniería básica en un conglomerado.

8.1.23 Desactivación: Verificación por puntos de comprobación de un conglomerado, seguida de la supresión del conglomerado.

8.1.24 Clonación («cloning»): Instanciación de un conglomerado a partir de un punto de comprobación de conglomerado.

8.1.25 Recuperación: Clonación de un conglomerado tras el fallo o la supresión de un conglomerado.

8.1.26 Reactivación: Clonación de un conglomerado tras su desactivación.

8.1.27 Traslado: Traspaso de un conglomerado a una cápsula diferente.

8.2 Reglas de estructuración

Una especificación de ingeniería define la infraestructura requerida para soportar la distribución funcional de un sistema ODP mediante:

- la identificación de las funciones ODP requeridas para gestionar la distribución, comunicación, procesamiento y almacenamiento físicos;
- la identificación de los roles (o papeles) de los diferentes objetos de ingeniería que soportan las funciones ODP (por ejemplo, el núcleo).

Una especificación de ingeniería se expresa en términos de:

- una configuración de objetos de ingeniería, estructurada como conglomerados, cápsulas y nodos;
- las actividades que tienen lugar dentro de esos objetos de ingeniería;
- las interacciones de esos objetos de ingeniería.

Una identificación de ingeniería está constreñida por reglas del lenguaje de ingeniería. Estas reglas comprenden:

- reglas de canales (véase 8.2.1), reglas de referencia de interfaz (véase 8.2.2), reglas de vinculación distribuida (véase 8.2.3) y reglas de reubicación (véase 8.2.4) para la provisión de interacciones transparentes a la distribución entre objetos de ingeniería;
- reglas de conglomerados (véase 8.2.5), reglas de cápsulas (véase 8.2.6) y reglas de nodos (véase 8.2.7) que rigen la configuración de objetos de ingeniería;
- reglas de fallos (véase 8.2.9).

8.2.1 Reglas de canales

Un canal soporta la interacción transparente a la distribución entre objetos de ingeniería. Incluye, por ejemplo:

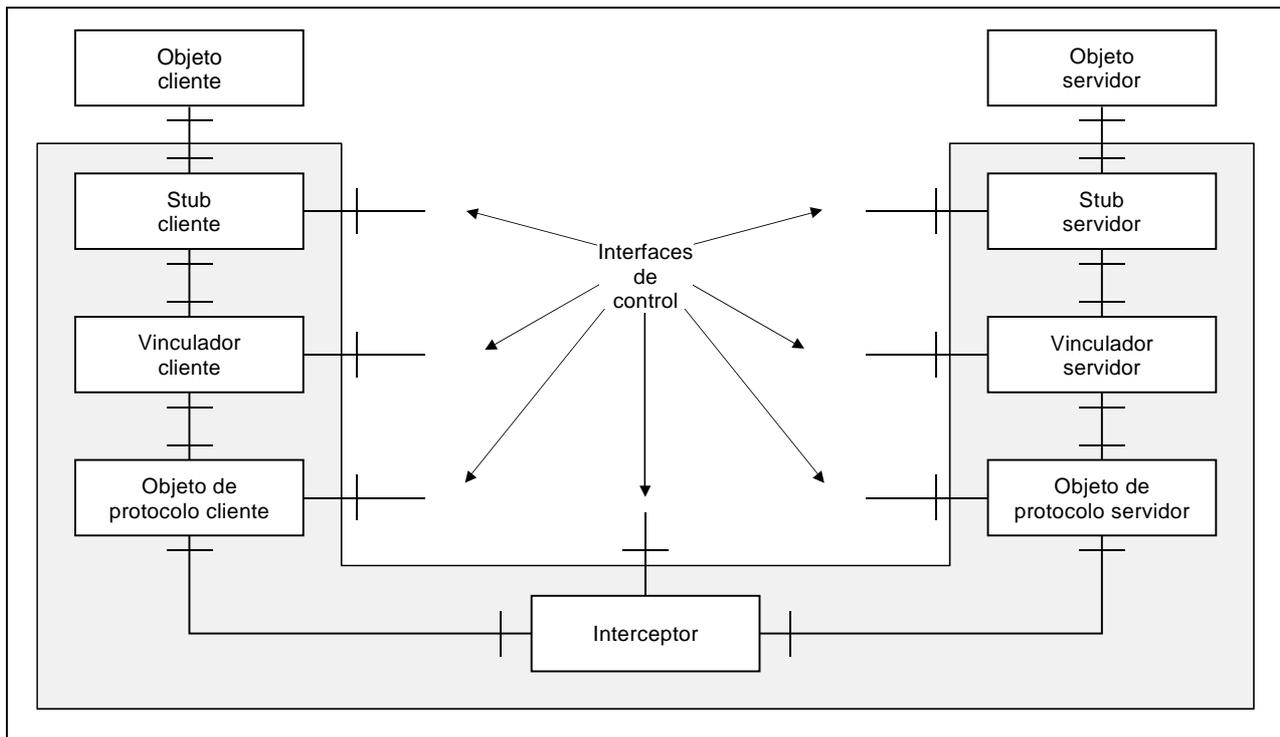
- la ejecución de operaciones entre un objeto cliente y un objeto servidor;
- un grupo de objetos que envían información mediante multidifusión a otros grupos de objetos;
- interacciones de trenes que comprenden múltiples objetos productores y múltiples objetos consumidores.

La interacción entre objetos de ingeniería entraña la transferencia de todas las informaciones siguientes o de algunas de ellas:

- referencias de interfaz de ingeniería;
- plantillas de conglomerados;
- datos.

Un canal es una configuración de stubs, vinculadores, objetos de protocolo e interceptores que interconectan un conjunto de objetos de ingeniería. La configuración es un gráfico acíclico con objetos stub en los vértices más exteriores como se ilustra en la Figura 2 y en la Figura 3. Cada trayecto a través del gráfico entre objetos stub está constituida por una secuencia, o bien de:

- un vinculador, un objeto de protocolo, un objeto de protocolo y un vinculador; o
- un vinculador, un objeto de protocolo, un interceptor, un objeto de protocolo y un vinculador.



TISO6300-95/d01

Figura 2 – Ejemplo de un canal cliente/servidor básico

El comportamiento de un canal con respecto a una configuración de canal o a la gestión de la calidad de servicio se controla a través de interfaces de control de stubs, vinculadores, objetos de protocolo e interceptores. Estos interfaces de control son facultativos.

NOTAS

1 Los stubs, vinculadores, objetos de protocolo e interceptores en un canal pueden tener vinculaciones (locales o distribuidas) con otros objetos de ingeniería fuera del canal que proporcionan, por ejemplo, funciones de reubicación o coordinación.

2 Según la clase de transformación que se efectúa, un interceptor puede descomponerse en stubs, vinculadores, objetos de protocolo y objetos de ingeniería básicos, que reflejan la estructura de canal.

Los objetos en un canal pueden ser objetos de ingeniería básicos soportados, ellos mismos, por canales.

8.2.1.1 Stubs

Los objetos de ingeniería básicos que interactúan con canales están vinculados localmente a stubs. En un canal, los stubs proporcionan la conversión de los datos transportados por interacciones. Los stubs pueden aplicar controles y llevar registros (por ejemplo, para seguridad y contabilidad). Los stubs pueden interactuar con objetos de ingeniería fuera de los canales (por ejemplo, con una función de seguridad) si es necesario. Un stub en un canal tiene una interfaz para uso por el objeto de ingeniería básica que soporta y una interfaz para la interacción con un vinculador. Puede también tener una interfaz de control.

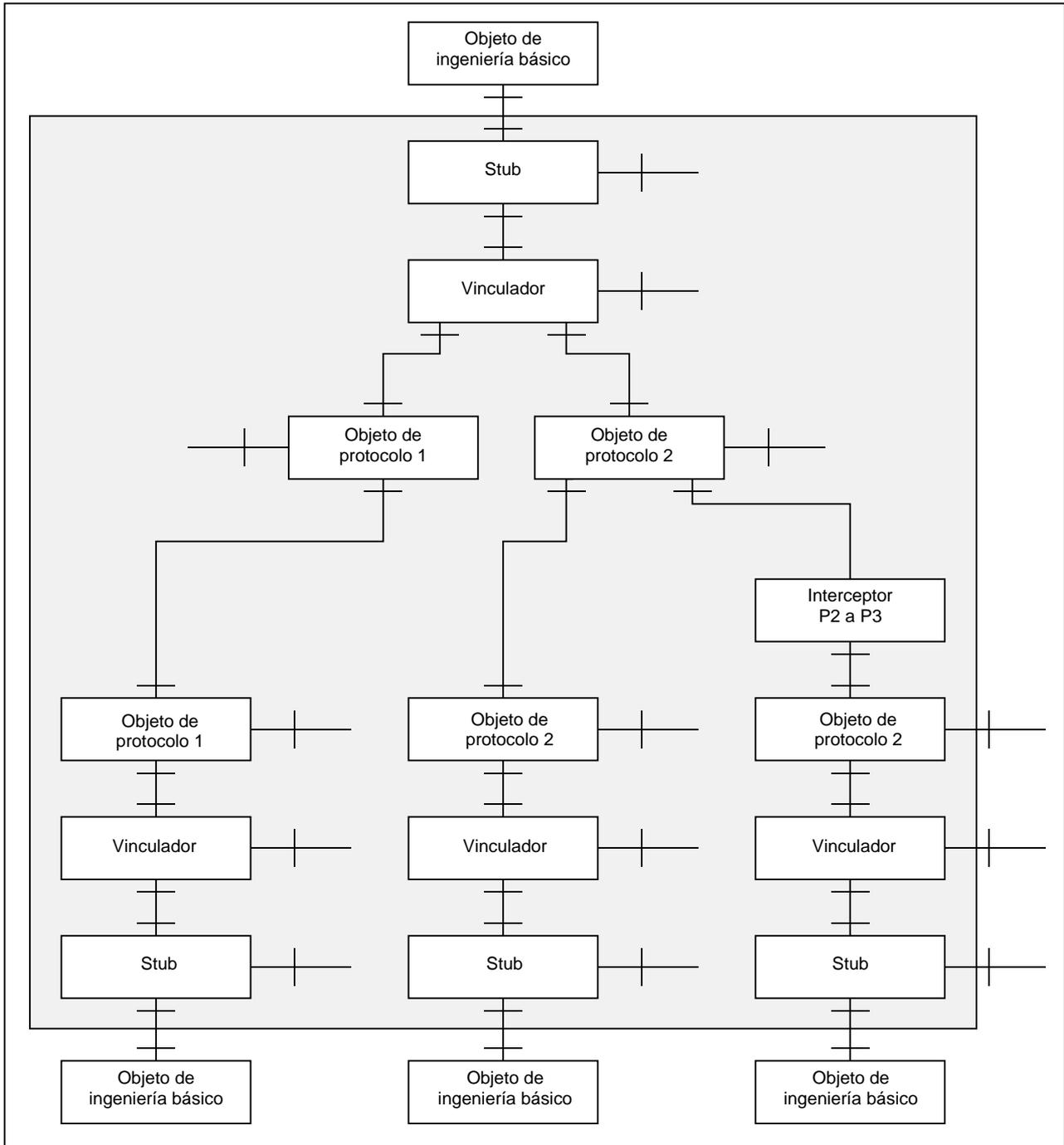
Cuando los stubs interconectados utilizan diferentes sintaxis de transferencia tiene que haber, en el trayecto entre ellos, un interceptor que pueda transformar datos de una sintaxis a otra.

Un stub puede tener una de las formas siguientes:

- específico de la *instancia de interfaz* de objeto de ingeniería básico al que está vinculado;
- específico del *tipo de interfaz* de la interfaz de objeto de ingeniería básico a la que está vinculado (por tanto, el stub puede ser compartido por varios canales del mismo tipo);
- genérico (es decir, no específico de un solo tipo de canal); estos stubs pueden ser compartidos por varios canales de diferentes tipos.

NOTAS

- 1 Cuando tiene la forma a), las interacciones entre el objeto de ingeniería y el stub sólo transportan datos de interacción (por ejemplo, nombres de operaciones y parámetros en el caso de una invocación). El stub actúa como un representante local para cada uno de los objetos de ingeniería básicos vinculados al canal.
- 2 Cuando tiene la forma b), las interacciones incluirán un identificador para el canal que se va a utilizar.
- 3 Cuando tiene la forma c), las interacciones incluirán un identificador y un tipo para el canal que va a utilizarse, de modo que el stub pueda asegurar que los datos de interacción son compatibles con el tipo de canal.



TISO6310-95/d02

Figura 3 – Ejemplo de un canal de múltiples puntos extremos

8.2.1.2 Vinculadores

Los vinculadores en un canal controlan la integridad de extremo a extremo de ese canal. Cuando es necesario, los vinculadores proporcionan transparencia de reubicación detectando fallos de la comunicación y reparando vinculaciones distribuidas rotas. Los vinculadores en un canal pueden interactuar con objetos de ingeniería fuera del canal para obtener datos adicionales necesarios para efectuar su función (por ejemplo, interactuar con un reubicador para obtener datos de ubicación). Un vinculador en un canal tiene por lo menos una interfaz para la interacción con un stub y una o más interfaces para la interacción con objetos protocolo. Un vinculador puede tener una interfaz de control. Si está presente, la interfaz de control permite efectuar cambios en la configuración del canal y suprimir el canal.

8.2.1.3 Objetos de protocolo

Los objetos de protocolo proporcionan funciones de comunicación. Los objetos de protocolo pueden interactuar con objetos de ingeniería fuera de un canal (por ejemplo, con funciones de directorio) para obtener la información que necesiten. Un objeto de protocolo tiene una interfaz para la interacción con un vinculador y por lo menos una interfaz de comunicaciones para la interacción con otros objetos de protocolo (a través de interceptores, si es necesario). Un objeto de protocolo puede tener una interfaz de control. Cuando los objetos de protocolo en un canal son de tipo diferente, requieren un interceptor que proporcione la conversión de protocolo. Todos los objetos de protocolo en un dominio de comunicación pueden comunicar directamente utilizando facilidades del dominio de comunicaciones (estas facilidades están fuera del ámbito de este modelo de referencia).

En cualquier ubicación en el tiempo dada, un objeto de protocolo se identifica por su ubicación en el espacio, pero diferentes objetos de protocolo pueden ocupar la misma ubicación en el espacio en tiempos diferentes (por ejemplo, las direcciones de red pueden ser recicladas). Cuando los objetos de protocolo en un canal son del mismo tipo, pero se encuentran en dominios de comunicaciones disjuntos, son posibles conflictos de denominación (por ejemplo, unos nombres de interfaces de comunicaciones podrían ser ambiguos). En esta situación es necesario un interceptor para transformar nombres intercambiados en el proceso de establecimiento del canal y en el proceso de mantenimiento de la integridad del canal.

8.2.1.4 Interceptores

Un interceptor <x> en un canal está situado en la frontera entre dominios <x> y proporciona comprobaciones y transformaciones de interacciones que atraviesan la frontera entre los dominios <x>. En función de la frontera atravesada, los interceptores requieren diferentes informaciones para realizar su tarea. Algunos interceptores necesitarán saber los tipos de firma de las interfaces de los objetos de ingeniería básicos vinculados al canal en el que están situados (los interceptores), de modo que puedan interpretar las interacciones soportadas por el canal. Un interceptor tiene por lo menos dos interfaces de comunicación. Puede tener una interfaz de control.

8.2.2 Reglas de referencia de interfaz

A los fines de la vinculación distribuida, las interfaces de ingeniería se ubican en el espacio y en el tiempo por medio de referencias de interfaz de ingeniería. Las referencias de interfaz de ingeniería se definen con relación a un dominio de gestión de referencias de interfaz de ingeniería que determina la política para el contenido, atribución, rastreo y validación de referencias de interfaz de ingeniería dentro de ese dominio. Un dominio de gestión de referencias de interfaz de ingeniería está constituido por un conjunto de nodos. Los dominios de referencias de interfaz de ingeniería pueden federarse si sus políticas de gestión de referencias de interfaz de ingeniería no están en conflicto.

Una referencia de interfaz de ingeniería contiene información que permite establecer vinculaciones con interfaces de objetos de ingeniería. Esta información permite a los objetos núcleos crear canales y también permite a los vinculadores en los canales mantener una vinculación distribuida entre los objetos de ingeniería interconectados. La información en una referencia de interfaz de ingeniería puede adoptar la forma de:

- datos;
- identificadores para interfaces que dan acceso a esos datos;
- una combinación de datos e identificadores.

Para efectuar las vinculaciones se necesitan algunos o todos los datos siguientes:

- el tipo de interfaz de la interfaz referenciada;
- una plantilla de canal que describe los interceptores, objetos de protocolo, vinculadores y stubs que pueden seleccionarse cuando se configura un canal para que soporte la vinculación distribuida;
- la ubicación en el espacio y en el tiempo (por ejemplo, una dirección de red) de las interfaces de comunicación en las que puede iniciarse el proceso de vinculación;
- información que permita la detección y reparación de vinculaciones distribuidas invalidadas por la reubicación de un objeto de ingeniería.

Un dominio de gestión de referencias de interfaz puede dividirse en dos subdominios. En este caso, las referencias de interfaz dentro del dominio se organizan como un conjunto de conjuntos alternativos de información, uno para cada subdominio en el que habrá de permitirse la vinculación.

NOTA 1 – Si el núcleo que soporta la interfaz de ingeniería soporta diferentes protocolos, procesos de vinculación y sintaxis de transferencia, la referencia de interfaz de ingeniería indicará las combinaciones válidas que pueden seleccionarse en una vinculación distribuida determinada; diferentes vinculaciones podrían hacer diferentes selecciones.

NOTA 2 – Este modelo de referencia no prescribe el método para obtener, a partir de una referencia de interfaz de ingeniería, una plantilla de canal y la ubicación en el espacio y en el tiempo de interfaces relacionadas.

Las referencias de interfaz de ingeniería son atribuidas por núcleos, en interfaces que soportan la función de gestión de nodo (véase 12.1.3). La función de rastreo de referencias de interfaz de ingeniería (véase 13.9) rastrea referencias de interfaz de ingeniería para detectar interfaces de ingeniería que ya no están referenciadas. La función de reubicación (véase 14.3) registra la política para la revinculación con interfaces de objeto de ingeniería que han sido reubicadas y las referencias de ingeniería actualizadas para interfaces de ingeniería que han sido reubicadas. Estas tres funciones (gestión de nodos, rastreo de referencias de interfaz de ingeniería y reubicación) pueden coordinarse utilizando una función de organización de información compartida (véase 14.2).

Las referencias de interfaz de ingeniería son unívocas en el contexto de denominación de un dominio de gestión de referencias de interfaz de ingeniería. Para conseguir esta univocidad, los nodos en el dominio de gestión de referencias de interfaz de ingeniería tienen que atribuir referencias de interfaz de ingeniería de una manera coordinada. Las interfaces de ingeniería se atribuirán de una manera que impida que una referencia de interfaz de ingeniería señale una interfaz incorrecta, incluso en presencia de fallos y reubicación de interfaz. En el peor de los casos, una referencia de interfaz señalará una interfaz que no existe (por ejemplo, inmediatamente después de un fallo del objeto de ingeniería que soporta la interfaz).

NOTA 3 – En sistemas ODP en que la mayoría de las interfaces no cambian su ubicación, la gestión de referencias de interfaz puede optimizarse: el núcleo puede atribuir referencias de interfaz de ingeniería autónomamente; el tipo de canal y el identificador de interfaz de comunicaciones asociados con la interfaz pueden almacenarse y transmitirse dentro de la referencia de interfaz de ingeniería; la función de reubicación puede utilizarse para validar y actualizar referencias de interfaz de ingeniería para interfaces que hayan sido reubicadas.

Antes de utilizar una referencia de interfaz de ingeniería, el núcleo construye una plantilla de canal que define una configuración de stub, vinculador y objetos de protocolo adecuados para soportar interacciones en la interfaz. Además, el núcleo establece una estructura local suficiente para hacer posible la vinculación al interfaz, y asocia la plantilla y la estructura local con una interfaz de comunicaciones. La referencia de interfaz de ingeniería pone a disposición esta información.

Un interceptor que está situado en una frontera entre dominios de gestión de referencias de interfaz de ingeniería mantiene correspondencias entre referencias de interfaz de ingeniería en esos dominios. Cuando una referencia de interfaz de ingeniería o una plantilla de conglomerado que contiene referencias de interfaz de ingeniería atraviesan una frontera de dominio de referencias de interfaz de ingeniería, las referencias de interfaz de ingeniería pertinentes tienen que ser transformadas para que sean válidas en el nuevo dominio.

El intercambio de referencias de interfaz de ingeniería de un dominio de gestión de referencias de interfaz a otro sólo es posible cuando hay un procedimiento definido para establecer la correspondencia de las referencias y evitar así las ambigüedades.

8.2.3 Reglas de vinculación distribuida

El establecimiento de un canal requiere la creación de stubs, vinculadores, objetos de protocolo e interceptores adecuados. El establecimiento de un canal puede iniciarlo cualquier objeto de ingeniería, proporcionado por cada núcleo como una función de su interfaz de gestión de nodos. La vinculación distribuida comprende la interacción con los núcleos de los nodos en que están situadas las interfaces que habrán de ser vinculadas. El establecimiento del canal es parametrizado por una plantilla de canal y un conjunto de referencias de interfaz a cada una de las cuales se asigna un cometido particular en la plantilla de canal. La plantilla de canal será compatible con los tipos de canal indicados por las referencias de interfaz de ingeniería, para las interfaces que habrán de ser vinculadas. El núcleo para cada objeto que habrá de vincularse crea una configuración de stubs, vinculadores y objetos de protocolo en su nodo para soportar las interfaces del objeto que está siendo vinculado. Esto incluye la configuración de sus interfaces de control. Los objetos de protocolo que soportan el canal se conectan (posiblemente mediante interceptores) en sus interfaces de comunicación. La selección y configuración de stubs, vinculadores, objetos de protocolo e interceptores viene determinada por la plantilla de canal y los tipos de canales de las referencias de interfaz que intervienen. A cada objeto de ingeniería básico vinculado por el canal se asigna un identificador de punto extremo de vinculación para cada interfaz que tiene con el canal. Los objetos de ingeniería básicos utilizan identificadores de punto extremo de vinculación para indicar en cuál de sus interfaces habrá de producirse una interacción distribuida.

NOTAS

1 Cualquier objeto de ingeniería puede establecer un canal, independientemente de que dicho objeto tenga o no una interfaz que vaya a ser vinculada por el canal.

2 Un objeto de ingeniería básico que inicia una vinculación distribuida requiere un conjunto de referencias de interfaz. Estas pueden obtenerse de cualquiera de las siguientes maneras:

- a) al inicializarse el objeto;
- b) por interacción entre el objeto iniciador y el núcleo como parte de la instanciación de las interfaces del objeto iniciador;
- c) mediante alguna cadena de interacciones con otros objetos de interés (por ejemplo, por paso de parámetros, o por comercio (importación/exportación) de parámetros).

3 Una plantilla de canal puede contener configuraciones alternativas que se apliquen en determinadas circunstancias. Por ejemplo, si los trayectos de comunicación son inseguros, podría requerirse la encriptación de stubs.

8.2.4 Reglas de reubicación

Los objetos de ingeniería pueden reubicarse como resultado de:

- reactivación y desactivación;
- verificación por puntos de comprobación y restablecimiento;
- traslado;
- funciones de gestión de dominio de comunicaciones (por ejemplo, cambio de un identificador de interfaz de comunicaciones).

NOTAS

1 Una interfaz de comunicaciones puede ver modificado su identificador como consecuencia del cambio de la dirección de red de un nodo.

2 Cuando se restablecen canales, podrían utilizarse stubs, vinculadores y objetos de protocolo que fueran diferentes de los utilizados antes de la reubicación. Por tanto, un identificador de interfaz de comunicaciones no necesariamente basta para identificar una interfaz de objeto de ingeniería.

La reubicación puede provocar el fallo de canales e invalidar referencias de interfaz de ingeniería. Los canales que han fallado pueden repararse si la actividad que cambia la ubicación de la interfaz de objeto de ingeniería notifica adecuadamente a la función de reubicación (véase 14.3).

Los vinculadores de un canal detectan las situaciones en que la reubicación ha invalidado el canal. Entonces, o bien los vinculadores colaboran para corregir la correspondencia entre referencias de interfaz de ingeniería y la estructura de canales (es decir, se requiere transparencia de reubicación, véase 16.6), o el canal falla. Cuando se requiere transparencia de reubicación, la información disponible a través de esa referencia de interfaz de ingeniería hace posible que los vinculadores utilicen la función de reubicación para determinar las nuevas ubicaciones de los objetos de ingeniería básicos que intervienen.

8.2.5 Reglas de conglomerados

Un conglomerado contiene un conjunto de objetos de ingeniería básicos asociados con un gestor de conglomerado. Cada miembro de un conglomerado puede tener una interfaz que soporte la función de gestión de objetos. Cada una de estas interfaces de gestión de objetos está vinculada al gestor de conglomerado del conglomerado en cuestión. Un objeto de ingeniería básico en un conglomerado está siempre vinculado a su núcleo, en una interfaz que proporciona la función de gestión de nodo, y a su gestor de conglomerado. Además, un objeto de ingeniería básico en un conglomerado puede estar vinculado a otros objetos de ingeniería básicos en el mismo conglomerado, o en otros conglomerados. Cada gestor de conglomerado en una cápsula está vinculado al gestor de cápsula para esa cápsula. Esta estructura se ilustra en la Figura 4.

Un conglomerado está siempre contenido en una sola cápsula. Un conglomerado es responsable de su propia seguridad, pero puede ser ayudado por funciones de seguridad. Toda función de seguridad de ayuda deberá ser proporcionada por un objeto en la misma cápsula en la que se encuentra ese conglomerado, o se ganará acceso a dicha función mediante interacciones seguras si está fuera de la cápsula. Los objetos de ingeniería en el mismo conglomerado pueden interactuar utilizando una vinculación local dentro del conglomerado, o una vinculación distribuida soportada por un canal. Los objetos de ingeniería en conglomerados diferentes interactúan utilizando vinculaciones distribuidas soportadas por canales.

NOTAS

1 No se excluyen interacciones que se realicen en puntos de referencia perceptual o de intercambio.

2 Aunque no se necesita un canal para soportar vinculaciones locales entre objetos dentro del mismo conglomerado, el identificador utilizado para efectuar invocaciones es de la misma clase que el utilizado entre objetos de ingeniería en diferentes conglomerados, y se denomina igualmente identificador de punto extremo de vinculación.

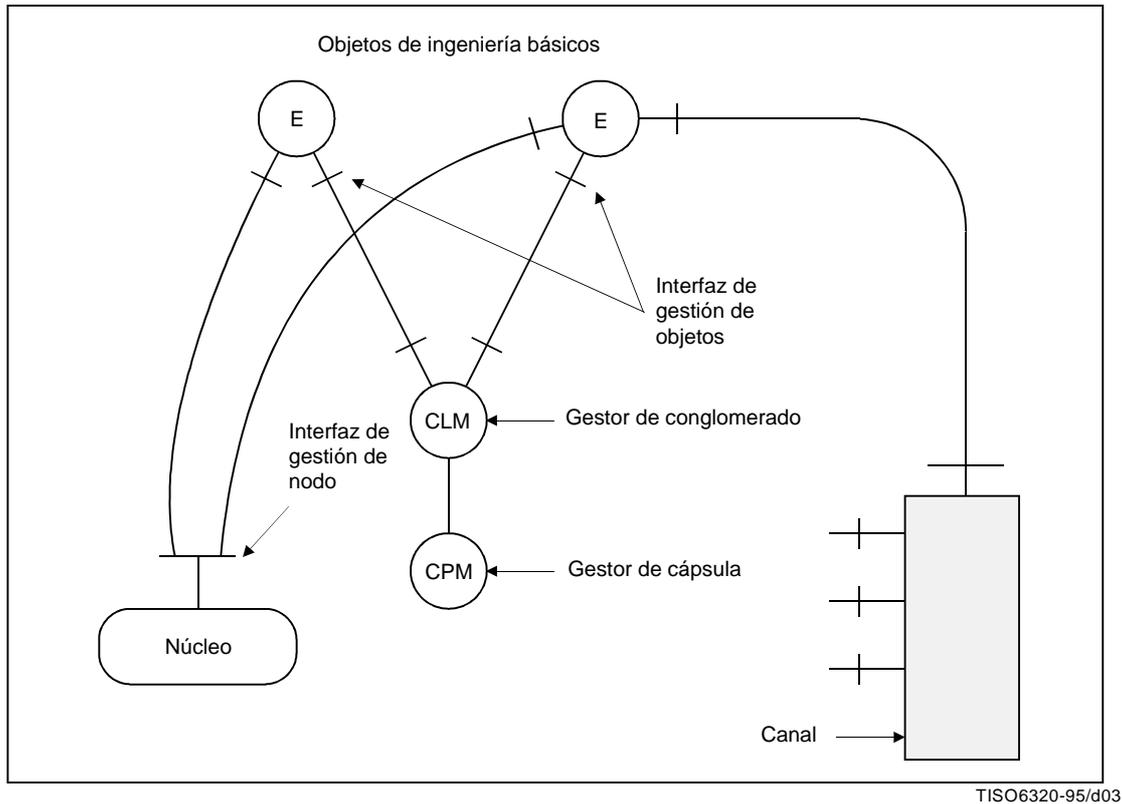


Figura 4 – Ejemplo de estructura que soporta un objeto de ingeniería básico

La instanciación del conglomerado (incluida la clonación como un caso especial) la efectúa un gestor de cápsula.

Si la plantilla es de un punto de comprobación de conglomerado, la instanciación (es decir, la clonación) permite que el nuevo conglomerado actúe como sustituto del conglomerado original del que se derivó la plantilla de conglomerado. Cuando se requiera por razones de transparencia de distribución, el proceso de clonación incluye el restablecimiento de toda vinculación distribuida que haya sido retenida por el conglomerado original.

Un conglomerado tiene asociado un gestor de conglomerado. El gestor de conglomerado proporciona la función de gestión de conglomerado. El gestor de conglomerado incorpora la política de gestión para los objetos de ingeniería en su conglomerado. La política de gestión de conglomerado pudiera conducir al gestor de conglomerado a interactuar con otras funciones ODP para la realización de actividades de gestión de conglomerado.

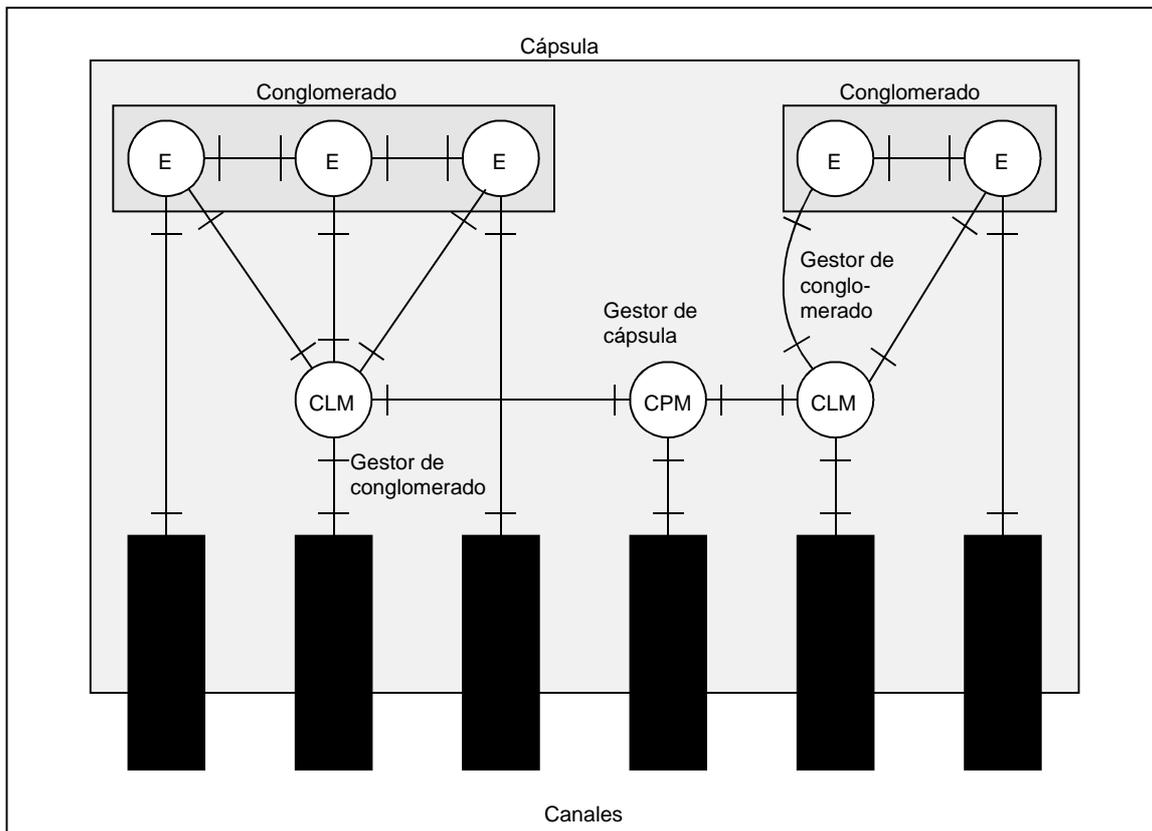
Un gestor de conglomerado ayuda a su gestor de cápsula en la gestión de las referencias de interfaz de ingeniería, de objetos. Esto puede requerir el acceso a la función de rastreo de las referencias de interfaz de ingeniería.

8.2.6 Reglas de cápsulas

Una cápsula consta de:

- uno o más conglomerados;
- gestores de conglomerado, uno para cada conglomerado en la cápsula;
- un gestor de cápsula al cual está vinculado cada uno de los gestores de conglomerado.

Los stubs, vinculadores y objetos de protocolo para un canal vinculado a una interfaz de un objeto de ingeniería básico dentro de un conglomerado en una cápsula pueden incluirse dentro de esa cápsula. Todos los objetos de ingeniería en una cápsula están vinculados a la misma interfaz de gestión de nodo. Los objetos de ingeniería en otras cápsulas están vinculados a diferentes interfaces de gestión de nodo. Una cápsula está contenida en un nodo. Una cápsula tiene un gestor de cápsula. El gestor de cápsula está vinculado, en una interfaz que proporciona la función de gestor de conglomerado, a cada gestor de conglomerado en la cápsula. Esta estructura se ilustra en la Figura 5 (en la que se ha hecho abstracción de los detalles del núcleo).



TISO6330-95/d04

Figura 5 – Ejemplo de la estructura de una cápsula

La instanciación de una cápsula la efectúa el núcleo utilizando una plantilla de cápsula que especifica la configuración inicial de los objetos de ingeniería en la cápsula, incluyendo conglomerados, gestores de conglomerado, stubs, vinculadores, protocolos y un gestor de cápsula.

Una cápsula es un contexto de denominación para identificadores de punto de extremo de vinculación. El modelo de referencia no requiere que estos identificadores sean válidos en un contexto más amplio. Se utilizan referencias de interfaz de ingeniería para comunicar conocimiento de interfaces de objeto de ingeniería entre cápsulas (para los fines de la vinculación).

El gestor de cápsula incorpora la política de gestión para los conglomerados en su cápsula. La política de gestión de cápsula podría conducir al gestor de cápsula a interactuar con otras funciones para realizar actividades de gestión de cápsula. El gestor de cápsula tiene una interfaz que proporciona la función de gestor de cápsula. Las estructuras que soportan la interacción entre gestores de conglomerado, gestores de cápsula y el núcleo dentro de un nodo son cuestiones de detalle de la implementación y están fuera del ámbito de este modelo de referencia.

8.2.7 Reglas de nodos

Un nodo consiste en un núcleo y un conjunto de cápsulas. Todos los objetos de ingeniería en un nodo comparten funciones comunes de procesamiento, almacenamiento y comunicación.

Un nodo pertenece a uno o más dominios de gestión de referencias de interfaz de ingeniería.

El núcleo proporciona un conjunto de interfaces de gestión de nodo, una para cada cápsula dentro del nodo.

La estructura de un nodo se ilustra en la Figura 6.

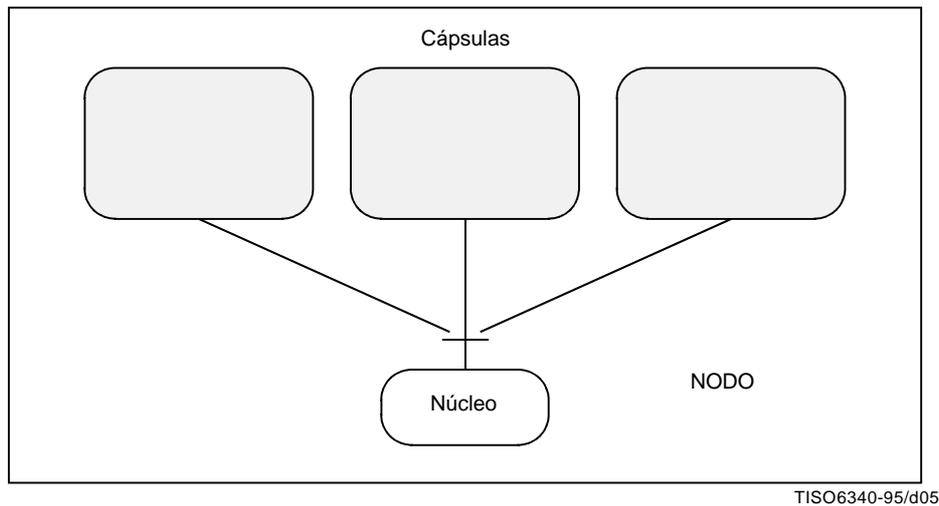


Figura 6 – Ejemplo de la estructura de un nodo

El procedimiento para la instanciación de los nodos está fuera del ámbito de este modelo de referencia, su resultado deberá ser el siguiente:

- introducción del núcleo del nodo y de sus funciones de procesamiento, almacenamiento y comunicación asociadas, incluida la introducción de funciones de gestión de nodo para hacer posible la vinculación distribuida de referencias de interfaz de ingeniería;
- introducción de cualquier función de comercio requerida por el proceso de instanciación;
- instanciación de cualquier canal requerido como parte de la configuración inicial del nodo (por ejemplo, para soportar objetos de ingeniería tales como un reubicador).

Un conjunto de objetos de protocolo introducidos durante la instanciación del nodo determina el conjunto inicial de dominios de comunicación a que pertenece el nodo. El núcleo proporciona la gestión de función de nodo e incorpora una política de gestión de nodo. Esta política podría conducir al núcleo a interactuar con otras funciones ODP para realizar sus actividades de gestión de nodo. Una cápsula es la unidad básica para la aplicación de la política de gestión de nodo; aunque un objeto individual puede utilizar la función de gestión de nodo, está sujeto a las políticas aplicables a su cápsula. Cápsulas distintas pueden estar sujetas a políticas diferentes de gestión de nodo.

8.2.8 Reglas de gestión de aplicaciones

La gestión del ciclo de vida (creación, traslado, desactivación, reactivación, verificación por punto de comprobación, fallo, recuperación y supresión) de conjuntos coordinados de conglomerados se rige por políticas de gestión específicas de aplicación. Las políticas de gestión de aplicación pueden aplicarse a conglomerados individuales o a conjuntos coordinados de conglomerados. Un conjunto de conglomerados gestionado forma un **dominio de gestión de aplicaciones**. La política de gestión de aplicaciones puede ser implementada por funciones de gestión específicas de aplicación que efectúan cambios utilizando los mecanismos proporcionados por las funciones de coordinación y gestión definidas en este modelo de referencia, por ejemplo la función de gestión de conglomerado.

Cuando proceda (es decir, según las transparencias de distribución que se apliquen) las funciones de gestión específica de aplicación pueden recibir notificaciones de eventos significativos que afectan a los conglomerados que ellas gestionan, y actuar en respuesta a esas notificaciones. Por ejemplo, informes de fallo de vinculación pueden conducir a la reactivación de un conglomerado, o informes de una carga de trabajo excesiva pueden conducir al traslado de conglomerados. Las peticiones y notificaciones relativas a un conglomerado en un dominio de gestión de aplicaciones puede hacer que las funciones de gestión específica de aplicación inicien acciones de ciclo de vida en otros conglomerados del dominio.

Los detalles específicos de la gestión de dominio de aplicaciones están fuera del ámbito de este modelo de referencia.

8.2.9 Reglas de fallos

Las reglas de fallos pueden agruparse en categorías según que comprendan conglomerados, cápsulas, nodos o dominios de comunicaciones. El análisis de los fallos se basa en el hecho de que:

- un fallo localizado en un conglomerado puede ser detectado por el gestor de ese conglomerado;
- un fallo localizado en una cápsula puede ser detectado por el gestor de esa cápsula;
- un fallo de un nodo puede ser detectado por objetos de protocolo en otros nodos con el que aquél está interconectado.
- el fallo de un dominio de comunicaciones puede ser detectado por objetos de protocolo en otros dominios de comunicaciones con el que aquél está interconectado.

NOTA – Existe una ambigüedad inherente en los sistemas de comunicaciones que puede impedir a un objeto de protocolo distinguir entre un fallo de comunicación y un fallo en un nodo distante.

8.3 Conformidad y puntos de referencia

Hay un punto de referencia programática en un punto de interacción entre un gestor de conglomerado y un objeto de ingeniería básico.

Hay un punto de referencia programática en un punto de interacción entre un objeto de ingeniería básico y un núcleo.

Hay un punto de referencia programática en un punto de interacción entre objetos de ingeniería básicos.

Puede haber un punto de referencia perceptual o de intercambio en una interfaz de un objeto de ingeniería básico.

Cuando la interacción entre objetos de ingeniería básicos se proporciona por un canal, para cada objeto de ingeniería básico que interviene hay un punto de referencia programática en los puntos de interacción entre esos objetos y los correspondientes stubs en el canal.

Dentro de la configuración de objetos que forman un canal, hay un punto de referencia programática en cada uno de los siguientes puntos de interacción, dentro del canal:

- entre stubs (haciendo abstracción de los vinculadores, objetos de protocolo e interceptores en el canal entre los stubs);
- entre stubs y vinculadores;
- entre vinculadores (haciendo abstracción de los objetos de protocolo e interceptores en el canal entre los vinculadores);
- entre vinculadores y objetos de protocolo;
- entre objetos de protocolo y otros objetos de protocolo dentro del mismo nodo (haciendo abstracción de los interceptores, si los hubiere, entre los objetos de protocolo),

y hay un punto de referencia de interfuncionamiento en cualquier punto de interacción entre objetos de protocolo y otros objetos de protocolo o interceptores en nodos diferentes.

Las interfaces de control de stubs, vinculadores, objetos de protocolo e interceptores son puntos de referencia programática.

Cuando los objetos de ingeniería dentro de un canal interactúan con otros objetos de ingeniería (sea dentro o fuera de ese canal) a través de interfaces que no están dentro de ese canal, los puntos de referencia aplicables a esas interfaces se determinan por la aplicación recursiva de estas reglas.

Definiendo la conformidad en puntos de referencia de interfuncionamiento se hace posible el interfuncionamiento de sistemas.

Definiendo la conformidad en puntos de referencia programática se hace posible la portabilidad de objetos de ingeniería entre sistemas.

La conformidad de objetos de ingeniería individuales en los puntos de referencia programática no garantiza por sí sola que el objeto de ingeniería será portable a todos los sistemas o que interfuncionará con objetos de ingeniería concordantes vinculados a otros núcleos.

9 Lenguaje de tecnología

Una especificación de tecnología define la tecnología elegida para un sistema ODP.

9.1 Conceptos

El lenguaje de tecnología consta de los conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2 y de los conceptos definidos en la presente Recomendación | Norma Internacional que cumplen las reglas de estructuración indicadas en 9.2.

9.1.1 Norma implementable: Una plantilla para un objeto de tecnología.

9.1.2 Implementación: Un proceso de instanciación cuya validez puede estar sometida a prueba.

9.1.3 IXIT: Información suplementaria de implementación para pruebas (Implementation eXtra Information for Testing).

9.2 Reglas de estructuración

Una especificación de tecnología define la tecnología elegida para un sistema ODP en términos de:

- una configuración de objetos de tecnología, y
- los interfaces entre esos objetos.

Una especificación de tecnología:

- expresa cómo son implementadas las especificaciones para un sistema ODP;
- identifica especificaciones de tecnología relativas a la construcción de sistemas ODP;
- proporciona una taxonomía para esas especificaciones;
- especifica la información que deben suministrar los implementadores para la realización de las pruebas.

Cuando se aplica una especificación escrita en otro lenguaje de punto de vista, se construye una especificación de tecnología para dar una interpretación de los términos atómicos en la otra especificación de punto de vista.

La especificación de tecnología para una función ODP puede hacer referencia a las especificaciones de otras funciones ODP.

Una especificación de tecnología consiste en enunciados de que los objetos de tecnología son instancias de normas implementables nominadas.

La especificación de tecnología da un formulario (o «proforma») para la IXIT de conformidad indicando el conjunto de plantillas y los nombres descriptivos requeridos para todos los puntos de referencia necesarios.

Todas las normas implementables se introducen por referencia a otras especificaciones. El lenguaje de tecnología no define ninguna otra regla que constriña el comportamiento de objetos de tecnología, o para la construcción de normas implementables.

9.3 Conformidad y puntos de referencia

El lenguaje de tecnología se utiliza para afirmar la certeza de que los objetos de tecnología son instancias de normas implementables; estas normas contendrán, por lo general, enunciados de conformidad.

10 Reglas de consistencia (o coherencia)

Un conjunto de especificaciones de un sistema ODP escritas en lenguajes de punto de vista diferentes no deben contener enunciados contradictorios entre sí (véase 4.2.2), es decir, deben ser mutuamente consistentes (coherentes). Por tanto, una especificación completa de un sistema incluye enunciados de correspondencias entre términos y construcciones de lenguaje que relacionan una especificación de punto de vista con otra especificación de punto de vista, mostrando que se satisface el requisito de la consistencia. La exigencia mínima de consistencia en un conjunto de especificaciones para un sistema ODP es que las especificaciones deben respetar las correspondencias definidas en este modelo de referencia y las definidas en el propio conjunto de especificaciones. Este modelo de referencia no declara correspondencias

genéricas entre cada par de lenguajes de punto de vista. Esta cláusula se limita a la especificación de correspondencias entre una especificación computacional y una especificación de información, y entre una especificación computacional y una especificación de ingeniería. En cada uno de estos dos casos, las correspondencias se expresan como relaciones de interpretación que enlazan términos en un lenguaje de punto de vista con términos en otro lenguaje de punto de vista. Se necesitará un conjunto de especificaciones basadas en este modelo de referencia, en general para relacionar todas las especificaciones de punto de vista.

La clave para la consistencia es la idea de correspondencias entre especificaciones, es decir, un enunciado de que algunos términos o estructuras de una especificación corresponden a términos y especificaciones en una segunda especificación. Se pueden establecer correspondencias entre dos especificaciones diferentes en un solo lenguaje o en dos lenguajes diferentes. Los enunciados de correspondencia entre dos lenguajes implican correspondencias equivalentes entre cualquier par de especificaciones expresadas en esos lenguajes.

El análisis de consistencia depende de la aplicación de técnicas específicas de consistencia. La mayoría de estas técnicas se basan en comprobaciones de clases particulares de consistencia, por lo que no pueden probar una consistencia absoluta. Una forma de consistencia implica un conjunto de reglas de correspondencia para dirigir una transformación de un lenguaje a otro. Así, dada una especificación S_1 en un lenguaje de punto de vista L_1 y una especificación S_2 en un lenguaje de punto de vista L_2 , donde S_1 y S_2 especifican ambos el mismo sistema, se puede aplicar una transformación T a S_1 y obtener como resultado una nueva especificación $T(S_1)$ en el lenguaje de punto de vista L_2 , que puede ser comparada directamente con S_2 para verificar, por ejemplo, la compatibilidad en comportamiento entre objetos o configuraciones de objetos que se pretende son equivalentes.

10.1 Correspondencias de especificaciones computacionales y de especificaciones de información

Este modelo de referencia no prescribe correspondencias exactas entre objetos de información y objetos computacionales. En particular, no todos los estados de una especificación computacional tienen que corresponder con estados de una especificación de información. Pueden existir estados computacionales transitorios dentro de piezas de comportamiento computacional que son simplificadas como transiciones atómicas en la especificación de información.

Cuando un objeto de información corresponde a un conjunto de objetos computacionales, los esquemas estático e invariante de un objeto de información corresponden a posibles estados de los objetos computacionales. Todo cambio en el estado de un objeto de información corresponde o bien a algún conjunto de interacciones entre objetos computacionales o a una acción interna de un objeto computacional. Los esquemas invariante y dinámico del objeto de información corresponden al comportamiento y al contrato de entorno de los objetos computacionales.

NOTA – Si el concepto de interfaz de información se utiliza en una especificación de información, no hay necesariamente una correspondencia entre una interfaz de información y cualquier interfaz computacional.

10.2 Correspondencias de especificaciones de ingeniería y de especificaciones computacionales

Cada objeto computacional que no es un objeto de vinculación corresponde a un conjunto de uno o más objetos de ingeniería básicos (y los canales que los conecten). Todos los objetos de ingeniería básicos en el conjunto corresponden solamente a ese objeto computacional.

A menos que intervengan transparencias que replican objetos, cada interfaz computacional corresponde exactamente a una interfaz de ingeniería, y esta interfaz de ingeniería corresponde solamente a aquel interfaz computacional.

NOTA 1 – La interfaz de ingeniería es soportada por uno de los objetos de ingeniería básicos, que corresponde al objeto computacional que soporta la interfaz computacional.

Cuando intervienen transparencias que replican objetos, cada interfaz computacional de los objetos que están siendo replicados corresponde a un conjunto de interfaces de ingeniería, uno para cada uno de los objetos de ingeniería básicos resultantes de la replicación. Cada una de estas interfaces de ingeniería corresponde solamente a la interfaz computacional original.

Cada interfaz computacional se identifica por cualquier miembro de un conjunto de uno o más identificadores de interfaz computacional. Cada interfaz de ingeniería se identifica por cualquier miembro de un conjunto de una o más referencias de interfaz de ingeniería. Así, puesto que una interfaz computacional corresponde a una interfaz de ingeniería, un identificador para una interfaz computacional puede representarse unívocamente por una referencia de interfaz de ingeniería tomada del conjunto correspondiente.

Cada vinculación computacional (ya se trate de vinculaciones primitivas o de vinculaciones compuestas con objetos de vinculación asociados) corresponde o bien a una vinculación local de ingeniería o a un canal de ingeniería. Esta vinculación local de ingeniería o canal de ingeniería corresponde solamente a esa vinculación computacional. Si la vinculación computacional soporta operaciones, la vinculación o canal de ingeniería local deberá soportar el intercambio de por lo menos:

- nombres de firma computacional;
- nombres de operación computacional;
- nombres de terminación computacional;
- parámetros de invocación y terminación (incluidos identificadores de interfaz computacional y firmas de interfaz computacional).

A menos que intervengan transparencias que replican objetos, cada interfaz de control de objeto de vinculación computacional tiene una interfaz de ingeniería correspondiente y existe una cadena de interacciones de ingeniería que enlaza esa interfaz con cualesquiera stubs, vinculadores, objetos de protocolo o interceptores que serán controlados para el soporte de la vinculación computacional.

NOTA 2 – El conjunto de interfaces de control que interviene depende del tipo del objeto de vinculación.

Cada interacción computacional corresponde a alguna cadena de interacciones de ingeniería, que comienza y termina por una interacción que comprende uno o más objetos de ingeniería básicos correspondientes a los objetos computacionales que interactúan.

Cada señal computacional corresponde o bien a una interacción en una vinculación local de ingeniería o a una cadena de interacciones de ingeniería que proporciona la visión consistente necesaria de la interacción computacional.

Las prescripciones en la cláusula 16 especifican correspondencias adicionales.

NOTA 3 – Los objetos de ingeniería básicos que corresponden a objetos computacionales diferentes pueden pertenecer al mismo conglomerado.

NOTA 4 – En un lenguaje computacional enteramente basado en objetos, los datos se representan como tipos de datos abstractos (es decir, como interfaces con objetos computacionales).

NOTA 5 – Los parámetros de interfaz computacional (incluidos los de tipos de datos abstractos) pueden pasarse por referencia; tales parámetros corresponden a referencias de interfaz de ingeniería.

NOTA 6 – Los parámetros de interfaz computacional (incluidos los de tipos de datos abstractos) pueden pasarse trasladando o replicando el objeto que soporta la interfaz. En el caso de traslado, estos parámetros corresponden a plantillas de conglomerado.

NOTA 7 – Si el estado abstracto de un objeto computacional que soporta un parámetro de interfaz es invariante, el objeto puede ser reproducido por clonación en lugar de ser trasladado.

NOTA 8 – Las plantillas de conglomerado pueden representarse como tipos de datos abstractos, por lo que basta con correspondencias estrictas entre parámetros computacionales y referencias de interfaz de ingeniería. La utilización de plantillas de conglomerados o datos constituyen importantes optimizaciones de la ingeniería, por lo que no se excluyen.

11 Funciones ODP

Las funciones ODP definidas en esta Recomendación | Norma Internacional son las funciones que son fundamentales o frecuentemente aplicables a la construcción de sistemas ODP.

Las especificaciones de funciones ODP individuales pueden combinarse para formar especificaciones de componentes de sistemas ODP. La identificación de estos componentes es una cuestión que incumbe al proceso de normalización y no se especifica en este modelo de referencia, en el que descripciones de funciones se esbozan utilizando conceptos de la Rec. UIT-T X.902 | ISO/CEI 10746-2.

Algunas descripciones de funciones en este modelo de referencia introducen objetos como una construcción que simplifica el modelado. Dichas descripciones no definen la estructura necesaria en una implementación, a menos que impongan constricciones explícitas sobre la distribución de esos objetos.

Las funciones definidas en este modelo de referencia se indican a continuación. Las que forman parte integrante del lenguaje computacional se señalan con «*», y las que forman parte integrante del lenguaje de ingeniería se señalan con un «+»:

- a) *Funciones de gestión*
 - 1) función de gestión de nodo+;
 - 2) función de gestión de objeto+;

- 3) función de gestión de conglomerado⁺;
 - 4) función de gestión de cápsula⁺;
- b) *Funciones de coordinación*
- 1) función de notificación de evento;
 - 2) función de verificación por punto de comprobación y recuperación;
 - 3) función de desactivación y reactivación;
 - 4) función de grupo;
 - 5) función de replicación;
 - 6) función de traslado;
 - 7) función de rastreo de referencia de interfaz de ingeniería⁺;
 - 8) función de transacción.
- c) *Funciones de depositario*
- 1) función de almacenamiento;
 - 2) función de organización de información;
 - 3) función de reubicación;
 - 4) función de depositario de tipo;
 - 5) función de comercio⁺*
- d) *Funciones de seguridad*
- 1) función de control de acceso;
 - 2) función de auditoría de seguridad;
 - 3) función de autenticación;
 - 4) función de integridad;
 - 5) función de confidencialidad;
 - 6) función de no repudio;
 - 7) función de gestión de claves.

12 Funciones de gestión

12.1 Función de gestión de nodo

La función de gestión de nodo controla las funciones de procesamiento, almacenamiento y comunicación dentro de un nodo.

La función de gestión de nodo la proporciona cada núcleo en uno o más interfaces de gestión de nodo.

Cada cápsula utiliza una interfaz de gestión de nodo distinta de las interfaces de gestión de nodo utilizadas por otras cápsulas en el mismo nodo.

La función de gestión de nodo:

- gestiona hilos;
- gana acceso a relojes y gestiona temporizadores;
- crea canales y ubica interfaces;

Dentro de la arquitectura definida por este modelo de referencia, todas las otras funciones utilizan la función de gestión de nodo.

12.1.1 Gestión de hilos

Las interfaces de gestión de nodo proporcionan funciones para establecer ramificaciones permanentes y temporales de hilos dentro de una cápsula y unir, demorar y sincronizar los hilos dentro de una cápsula.

12.1.2 Acceso a relojes y gestión de temporizadores

Las interfaces de gestión de nodo proporcionan funciones para determinar la hora (el tiempo) actual en un dominio de gestión de reloj especificado y arrancar, supervisar y cancelar temporizadores.

12.1.3 Creación de canales y ubicación de interfaces

Las interfaces de gestión de nodo proporcionan funciones para:

- a) permitir la vinculación entre un objeto de ingeniería dentro de una cápsula y una instancia de la función de comercio;
- b) poner a disposición una interfaz de ingeniería para la vinculación de objetos en otras cápsulas;
- c) establecer una vinculación entre un objeto de ingeniería dentro de una cápsula y un conjunto de otros objetos de ingeniería (identificados por referencias de interfaz de ingeniería);
- d) a partir de una determinada referencia de interfaz de ingeniería, determinar el tipo de canal y la interfaz de comunicación que implica.

NOTAS

- 1 La utilización de las interacciones b) y c) se explica en 8.2.3.
- 2 La interacción d) da a conocer el contenido de información de la referencia de interfaz, lo que le permite almacenarlo o transformarlo para uso en un dominio de gestión de referencia de interfaz diferente.

La interacción b) (poner a disposición una interfaz para la vinculación a objetos en otras cápsulas) comprende las siguientes acciones:

- asignación de una referencia de interfaz de ingeniería dentro de un dominio de gestión de referencias de interfaz de ingeniería nominado;
- asignación de una interfaz de comunicaciones a través de la cual podrán establecerse vinculaciones con la interfaz;
- asignación de un tipo de canal a la interfaz.

12.1.4 Instanciación de plantilla de cápsula y supresión de cápsula

Las interfaces de gestión de nodo proporcionan funciones para instanciar plantillas de cápsulas y suprimir cápsulas.

La instanciación de una plantilla de cápsula consta de los siguientes pasos:

- atribución de funciones de procesamiento, almacenamiento y comunicación a una nueva cápsula en el mismo nodo en que se encuentra el núcleo que proporciona la interfaz de gestión de nodo;
- creación de un gestor de cápsula para la nueva cápsula;
- creación de una interfaz de *gestión* de cápsula en el gestor de la nueva cápsula;
- producción de un identificador para la interfaz de gestión de cápsula;
- creación de una interfaz de *control* de cápsula para la nueva cápsula en el núcleo;
- producción de un identificador para la interfaz de control de cápsula.

La interfaz de control de cápsula producida por la instanciación de la plantilla de cápsula permite la supresión de la cápsula (por ejemplo, en caso de fallo de su gestor).

La supresión de una cápsula entraña la supresión de todos los objetos de la cápsula.

12.2 Función de gestión de objeto

La función de gestión de objetos verifica objetos por puntos de comprobación, y suprime objetos.

Cuando un objeto pertenece a un conglomerado que puede ser desactivado, verificado por punto de comprobación o trasladado, el objeto deberá tener una interfaz de gestión de objeto en el que proporcione una o más de las siguientes funciones:

- verificación del objeto por punto de comprobación;
- supresión del objeto.

NOTAS

1 Así, las diferentes interfaces de gestión de objeto pueden tener diferentes tipos de interfaz, lo que depende de las funciones que soportan.

2 La verificación de un objeto por punto de comprobación da información adecuada para la incorporación en un punto de comprobación de conglomerado.

3 Cuando se suprime un objeto, se pueden suprimir también los stubs, vinculadores, objetos de protocolo e interceptores que soportan vinculaciones a ese objeto.

La función de gestión de objeto es utilizada por la función de gestión de conglomerado.

12.3 Función de gestión de conglomerado

La función de gestión de conglomerado verifica por punto de comprobación, recupera, traslada, desactiva o suprime conglomerados y la proporciona cada gestor de conglomerado en una interfaz de gestor de conglomerado, que comprende una o más de las siguientes funciones con respecto al conglomerado gestionado:

- modificación de la política de gestión del conglomerado (por ejemplo, para la ubicación de puntos de comprobación de su conglomerado, para la utilización de la función de reubicación con el fin de efectuar la reactivación o recuperación del conglomerado);
- desactivación del conglomerado;
- verificación del conglomerado por punto de comprobación;
- sustitución del conglomerado por un conglomerado nuevo instanciado a partir de un punto de comprobación de conglomerado (es decir, supresión seguida de recuperación);
- traslado del conglomerado a otra cápsula (mediante la función de traslado);
- supresión de conglomerado.

NOTA – Así, las diferentes interfaces de gestión de objeto pueden tener diferentes tipos de interfaz, lo que depende de las funciones que soportan.

El comportamiento de un gestor de conglomerado está constreñido por la política de gestión de su conglomerado. La verificación de un conglomerado por punto de comprobación y su desactivación sólo es posible si todos los objetos en el conglomerado tienen interfaces de gestión de objeto que soportan la función de verificación de objeto por punto de comprobación. La desactivación y la supresión de un conglomerado requieren que los objetos en el conglomerado soporten la supresión de objeto.

Dentro de la arquitectura definida por este modelo de referencia:

- la función de gestión de conglomerado es utilizada por la función de gestión de cápsula, la función de desactivación y reactivación, la función de punto de comprobación y recuperación, la función de traslado y la función de gestión de referencia de interfaz de ingeniería;
- la función de gestión de conglomerado utiliza la función de almacenamiento para mantener los puntos de comprobación.

12.3.1 Punto de comprobación de conglomerado

Un punto de comprobación de conglomerado contiene la información necesaria para restablecer un conglomerado e incluye los siguientes elementos:

- a) puntos de comprobación de los objetos en el conglomerado;
- b) la configuración de los objetos en el conglomerado;
- c) información suficiente para restablecer vinculaciones distribuidas que comprenden objetos en el conglomerado.

NOTA – Las referencias de interfaz de ingeniería forman una parte esencial de la información requerida para establecer vinculaciones. Un punto de comprobación de conglomerado contendrá todas las referencias de ingeniería que se obtienen de a) y c).

12.3.2 Supresión, desactivación y fallo de conglomerado

La supresión de un conglomerado entraña la supresión de todos los objetos en el conglomerado, la del gestor del conglomerado y la de cualesquiera objetos que sólo soporten el conglomerado o su gestor de conglomerado (por ejemplo, stubs y vinculadores). La desactivación de un conglomerado es coordinada por la función de desactivación y reactivación; produce un punto de comprobación del conglomerado, después de lo cual suprime el conglomerado y su estructura de soporte. El fallo de un conglomerado provoca la supresión de todos los objetos en el conglomerado y, en algunos casos, conduce a la supresión de la estructura de soporte del conglomerado.

12.3.3 Reactivación y recuperación de conglomerado

Un conglomerado desactivado puede ser reactivado desde uno de sus puntos de comprobación. La reactivación de un conglomerado tiene necesariamente que ser una función de gestión de cápsula porque el gestor de conglomerado se suprime como parte de la desactivación del conglomerado. Un conglomerado puede ser recuperado desde uno de sus puntos de comprobación. Si el gestor de conglomerado asociado no ha sido suprimido, puede iniciar la recuperación; si ha sido suprimido, la recuperación es una función de gestión de cápsula e incluye la creación de un gestor del nuevo conglomerado.

12.3.4 Traslado de conglomerado

El traslado de conglomerado consiste en la clonación de un conglomerado fuente en una cápsula de destino, seguida por la supresión del conglomerado fuente. Es coordinado por la función de traslado y es parametrizado por una interfaz de gestión de cápsula para la cápsula de destino, para el conglomerado trasladado.

12.4 Función de gestión de cápsula

La función de gestión de cápsula instancia conglomerados (incluida la recuperación y reactivación), verifica por punto de comprobación todos los conglomerados en una cápsula, desactiva todos los conglomerados en una cápsula y suprime cápsulas. Es proporcionada por cada gestor de cápsula en una interfaz de gestión de cápsula que comprende una o más de las siguientes funciones con respecto a la cápsula gestionada:

- instanciación (dentro de la cápsula) de una plantilla de conglomerado;
 NOTA 1 – Incluye reactivación y recuperación.
- desactivación de la cápsula por la desactivación de todos los conglomerados dentro de ella (mediante la función de gestión de conglomerado);
- verificación de la cápsula por punto de comprobación, para lo cual se verifican por punto de comprobación todos los conglomerados en la cápsula (mediante la función de gestión de conglomerado);
- supresión de la cápsula, para lo cual se suprimen todos los conglomerados contenidos en ella y, seguidamente, se suprime el gestor de cápsula para esa cápsula.

NOTA 2 – Así, las diferentes interfaces de gestión de objeto pueden tener diferentes tipos de interfaz, lo que depende de las funciones que soportan.

El comportamiento de un gestor de cápsula está constreñido por la política de gestión para su cápsula.

Dentro de la arquitectura definida por este modelo de referencia, la función de gestión de cápsula es utilizada por la función de desactivación y reactivación, la función de punto de comprobación y recuperación y la función de traslado.

12.4.1 Instanciación de plantilla de conglomerado

La instanciación de una plantilla de conglomerado es parametrizada por una plantilla de conglomerado y consta de los siguientes pasos:

- instanciación de un conglomerado a partir de una plantilla de conglomerado;
- introducción de un gestor de conglomerado para el nuevo conglomerado;
- producción de un identificador para una interfaz de gestión de conglomerado en el gestor del nuevo conglomerado;
- vinculación del nuevo conglomerado a otros objetos de acuerdo con las reglas del lenguaje de ingeniería y la información de vinculación en la plantilla de conglomerado.

NOTA – La reactivación de conglomerado y la recuperación de conglomerado son casos especiales de instanciación de conglomerado en los cuales la plantilla de conglomerado es un punto de comprobación de conglomerado.

Una plantilla de conglomerado puede contener información específica de un dominio. Si la plantilla ha de ser instanciada en otro dominio, hay que transformar esta información. En particular, hay que transformar las referencias de interfaz de ingeniería contenidas en el conglomerado si el conglomerado se está instanciando en un dominio de gestión de referencias de interfaz de ingeniería diferente.

12.4.2 Supresión de cápsula

La supresión de cápsula entraña la supresión del gestor de cápsula y puede conducir a la supresión de stubs, vinculadores, objetos de protocolo o interceptores que soportaban objetos en la cápsula o su gestor.

13 Funciones de coordinación

13.1 Función de notificación de evento

La función de notificación de evento registra y pone a disposición historias de eventos.

13.1.1 Conceptos

13.1.1.1 Historia de evento: Objeto que representa acciones significativas.

13.1.2 Reglas

Los productores de eventos interactúan con la función de notificación de evento para crear historias de eventos. La función de notificación de evento notifica a los objetos consumidor de evento la disponibilidad de historias de eventos.

La función de notificación de evento soporta uno o más tipos de historia de evento e incorpora una política de notificación de eventos que determina el comportamiento de la función, en particular:

- qué objetos pueden crear historias de eventos;
- a qué objetos se notifica la creación de una nueva historia de evento;
- los medios por los cuales se producen esas notificaciones;
- los requisitos de persistencia y estabilidad de las historias de eventos;
- las relaciones de ordenación de las interacciones entre un objeto productor de evento y un objeto consumidor de evento.

Un consumidor de evento interactúa con la función de notificación de evento para registrar, con fines de notificación, las nuevas historias de eventos. Según la política de notificación de eventos, la interacción puede:

- establecer vinculaciones con historias de eventos actualmente disponibles;
- permitir la comunicación sobre historias de eventos creadas subsiguientemente a la interacción.

NOTA – Historias de eventos con requisitos estrictos de persistencia y estabilidad pueden ser soportadas mediante las funciones de transacción y de replicación. Las notificaciones de ordenación y multidifusión pueden ser soportadas mediante la función de grupo.

13.2 Función de punto de comprobación y recuperación

La función de punto de comprobación y recuperación coordina la verificación por punto de comprobación y la recuperación de conglomerados que han fallado.

La función de punto de comprobación y recuperación incorpora políticas que determinan:

- cuándo deben verificarse los conglomerados por punto de comprobación;
- cuándo deben recuperarse los conglomerados;
- dónde deben recuperarse los conglomerados;
- dónde deben almacenarse los puntos de comprobación;
- qué punto de comprobación se recupera.

La verificación por punto de comprobación y la recuperación de conglomerados están sujetas a cualquier política de seguridad asociada con esos conglomerados, en particular, las relativas a la determinación del lugar en que deberá almacenarse el punto de comprobación y del lugar en que deberá recuperarse el punto de comprobación.

Dentro de la arquitectura definida por este modelo de referencia, la función de punto de comprobación y recuperación utiliza la función de gestión de conglomerado y la función de gestión de cápsula.

13.2.1 Verificación por punto de comprobación

La verificación por punto de comprobación incumbe a la función de gestión de objeto y a la función de gestión de conglomerado. La verificación de un conglomerado por punto de comprobación la coordina su gestor de conglomerado: primeramente, el gestor de conglomerado utiliza la función de gestión de objeto para obtener un punto de comprobación de cada objeto en el conglomerado; a partir de estos puntos de comprobación de objetos, el gestor de conglomerado construye un punto de comprobación de conglomerado que se hace seguidamente persistente utilizando la función de almacenamiento.

En función de la política de verificación por punto de comprobación, la verificación por punto de comprobación de un conglomerado puede conducir a una verificación por punto de comprobación de otros conglomerados que participan en actividades comunes con el conglomerado verificado por punto de comprobación, de acuerdo con las siguientes reglas de consistencia:

- El conglomerado inicial deberá encontrarse en un estado consistente antes de ser verificado por punto de comprobación.
- Deberá haber consistencia entre todos los puntos de comprobación de conglomerado de los diversos conglomerados que se están verificando conjuntamente por punto de comprobación (por ejemplo, todos los puntos de comprobación reflejarán el mismo conjunto de interacciones realizadas entre los conglomerados).
- Cuando un conglomerado es verificado por punto de comprobación, todos los otros conglomerados que tienen constricciones de punto de comprobación con respecto al conglomerado en cuestión deberán ser verificados por punto de comprobación. Esta regla se aplicará recursivamente para obtener un conjunto cerrado de conglomerados que pueden ser verificados por punto de comprobación consistentemente.

13.2.2 Recuperación

Un conglomerado puede ser recuperado:

- o bien en la cápsula a partir de la cual fue anteriormente verificado por punto de comprobación, o
- en otra cápsula (por ejemplo, cuando la cápsula en que se ha producido la verificación por punto de comprobación ha fallado subsiguientemente).

La recuperación de un conglomerado incumbe al gestor de conglomerado para ese conglomerado o, en ausencia de dicho gestor de conglomerado, al gestor de cápsula. La función de punto de comprobación y recuperación interactúa con el gestor de conglomerado o con el gestor de cápsula, según proceda, para instanciar el punto de comprobación de conglomerado. Antes de recuperar un conglomerado, la función de punto de comprobación y recuperación deberá asegurar que el conglomerado ha sido suprimido (por ejemplo, como consecuencia de un fallo). Los objetos vinculados a conglomerados recuperados deberán poder detectar que el conglomerado ha sido recuperado desde un punto de comprobación (por ejemplo, de modo que puedan volver a efectuar las interacciones que ocurrieron después de tomado el punto de comprobación).

La recuperación de un conglomerado puede conducir a la recuperación de otros conglomerados, por ejemplo de los que forman un punto de comprobación conjunto con el conglomerado que ha sido recuperado.

13.3 Función de desactivación y reactivación

La función de desactivación y reactivación coordina la desactivación y reactivación de conglomerados. Incorpora políticas que determinan:

- cuándo deben desactivarse los conglomerados;
- dónde almacenarse el punto de comprobación asociado con una desactivación;
- cuándo deben reactivarse los conglomerados;
- qué punto de comprobación debe reactivarse (por ejemplo, el más reciente);
- dónde deben reactivarse los conglomerados.

La desactivación y reactivación de conglomerados están sujetas a cualquier política de seguridad asociada con esos conglomerados, en particular dentro de la arquitectura definida por este modelo de referencia, la función de desactivación y reactivación utiliza la función de gestión de objeto, la función de gestión de conglomerado y la función de gestión de cápsula. La función de desactivación y reactivación es utilizada por la función de traslado.

13.3.1 Desactivación

La desactivación de un conglomerado es una función de gestión de conglomerado y comprende los siguientes pasos:

- el gestor de conglomerado para el conglomerado en cuestión interactúa con cada objeto en su conglomerado para obtener un punto de comprobación que pueda utilizarse para establecer un punto de comprobación de conglomerado;
- el gestor de conglomerado hace persistente el punto de comprobación de conglomerado mediante la función de almacenamiento;
- el gestor de conglomerado suprime el conglomerado (y puede suprimirse a sí mismo).

13.3.2 Reactivación

La función de desactivación y reactivación reactiva un conglomerado utilizando la función de gestión de cápsula para instanciar un punto de comprobación del conglomerado en la cápsula de destino (e incluye la creación de un gestor de conglomerado para ese conglomerado). La cápsula de destino puede ser, o bien la cápsula desde la cual el conglomerado fue anteriormente desactivado, u otra cápsula (por ejemplo, para repartir equilibradamente la carga de la infraestructura entre varios nodos).

13.4 Función de grupo

La función de grupo proporciona el mecanismo necesario para coordinar las interacciones de objetos en una vinculación multipartita.

13.4.1 Conceptos

13.4.1.1 Grupo de interacción: Subconjunto de los objetos que participan en una vinculación gestionada por la función de grupo.

13.4.2 Reglas

Para cada conjunto de objetos que están vinculados conjuntamente en un grupo de interacción, la función de grupo gestiona:

- la **interacción**: decidiendo qué miembros del grupo participan en qué interacciones, de acuerdo con una política de interacción;
- la **colación**: obteniendo una visión consistente de las interacciones (incluidas las que han fallado), de acuerdo con una política de colación;
- la **ordenación**: asegurando que las interacciones entre los miembros del grupo estén correctamente ordenadas de acuerdo con una política de ordenación;
- la **membrecía** (pertenencia al grupo): tratando el fallo y recuperación de miembros, así como la adición y supresión de miembros de acuerdo con una política de pertenencia al grupo.

NOTA – El comportamiento del objeto de vinculación que enlaza miembros del grupo determina la manera de efectuar la interacción.

13.5 Función de replicación

La función de replicación («replication function») es un caso especial de la función de grupo en el cual los miembros de un grupo son compatibles en comportamiento (por ejemplo, porque son reproducciones exactas, o sea, «réplicas» de la misma plantilla de objeto). Esta función de replicación asegura que el grupo sea percibido, por otros objetos, como si fuera un solo objeto, al asegurar que todos los miembros participen en todas las interacciones, y que lo hagan en el mismo orden.

La política de pertenencia a un grupo de réplica puede permitir el aumento o la disminución del número de miembros de ese grupo de réplica. El aumento del tamaño del grupo de réplica produce el mismo efecto que si se hubiera creado un clon de un miembro del grupo y se hubiera añadido al grupo en una sola acción atómica.

Para aplicar una función de replicación a un conglomerado, los objetos que constituyen el conglomerado son replicados y configurados de modo que formen un conjunto de conglomerados idénticos. Los objetos correspondientes en cada uno de estos conglomerados replicados forman grupos de réplica. Por tanto, un conglomerado replicado es un conjunto coordinado de grupos de réplica.

La función de traslado utiliza la función de replicación.

13.6 Función de traslado

La función de traslado coordina el traslado de un conglomerado de una cápsula a otra. Utiliza la función de gestión de conglomerado y la función de gestión de cápsula, e incorpora políticas que determinan cuándo los conglomerados deben ser trasladados y dónde deben ser ubicados.

El traslado puede efectuarse por dos medios diferentes:

- replicación, o
- desactivación en una cápsula y subsiguiente reactivación en otra cápsula.

13.6.1 Replicación

El traslado de un conglomerado por medio de la función de replicación comprende la siguiente secuencia de acciones:

- el conglomerado antiguo se trata como un grupo de réplica de tamaño 1;
- en la cápsula de destino se crea una copia del conglomerado original, y un gestor de conglomerado;
- con los objetos de los dos conglomerados se forman grupos de réplica (de tamaño 2);
- los objetos en el conglomerado antiguo se suprimen en los grupos de réplica (después de esto, los grupos de réplica de tamaño 1);
- se suprime el conglomerado antiguo (y su gestor).

13.6.2 Desactivación y reactivación

El traslado de un conglomerado por desactivación y reactivación lo coordina el gestor del conglomerado, y comprende la desactivación del conglomerado en su ubicación antigua, y su subsiguiente reactivación en su nueva ubicación.

13.7 Función de transacción

13.7.1 Conceptos

13.7.1.1 Transacción: Actividad que conduce a un conjunto de cambios de estado de objeto de acuerdo con un esquema dinámico (y su esquema invariante constrictivo).

13.7.1.2 Acción de interés: En una transacción, acción que conduce a un cambio de estado significativo para esa transacción.

13.7.1.3 Visibilidad: Grado en que una transacción puede ganar acceso a un estado de objeto concurrentemente con otras transacciones.

13.7.1.4 Recuperabilidad: Grado en que los cambios de estado de objeto resultantes de transacciones fracasadas son anulados.

13.7.1.5 Permanencia: Grado en que los fallos pueden influir en los cambios de estado de objeto como consecuencia de transacciones ejecutadas.

13.7.2 Reglas

La función de transacción coordina y controla un conjunto de transacciones para alcanzar un nivel especificado de visibilidad, recuperabilidad y permanencia.

La función de transacción:

- interactúa con objetos para supervisar la concurrencia de acciones de interés, la anulación de los efectos de acciones de interés y la causalidad de acciones de interés;
- decide si las acciones de interés están en conflicto;
- interactúa con objetos para determinar el momento en que habrán de realizarse acciones de interés, para evitar conflictos;
- interactúa con objetos para anular los efectos de acciones de interés que se han realizado, con el fin de resolver conflictos.

Está sujeta a políticas que determinan:

- qué acciones son acciones de interés;
- qué acciones de interés están en conflicto;
- qué acciones de interés habrán de anularse para resolver conflictos.

13.8 Función de transacción ACID

La función de transacción ACID (*atomicity, consistency, isolation, durability*) es un caso especial de la función de transacción general en la cual:

- la visibilidad se especifica como el aislamiento de las transacciones, unas de las otras;
- la recuperabilidad se especifica como el requisito de que las transacciones sean atómicas;

- la permanencia se especifica como el requisito de que los cambios de estado que se produzcan como consecuencia de transacciones sean duraderos (es decir, estables);
- la consistencia se logra por una ejecución correcta de transacciones con las propiedades de atomicidad, aislamiento y durabilidad conformes con el esquema dinámico y el esquema invariante asociados.

Las acciones de interés para la función de transacción ACID son:

- comenzar la transacción;
- efectuar la transacción;
- abortar la transacción;
- ganar acceso al estado;
- modificar el estado;
- anular cualquiera de estas acciones.

Las políticas de transacción se expresan como reglas sobre las posibilidades de organizar en serie las transacciones.

13.9 Función de rastreo de referencia de interfaz de ingeniería

La función de rastreo de referencia de interfaz de ingeniería supervisa la transferencia de referencias de interfaz de ingeniería entre objetos de ingeniería en diferentes conglomerados para determinar cuándo la infraestructura asociada con interfaces de ingeniería deja de ser necesaria (es decir, cuando ningún objeto, en otro conglomerado, está en condiciones de vincularse a la interfaz referenciada).

Dentro su ámbito, la función de rastreo de referencia de interfaz de ingeniería mantiene:

- información sobre la posesión de referencias de interfaz de ingeniería;
- información sobre la existencia de interfaces.

La función de rastreo de referencia de interfaz de ingeniería:

- es notificada por stubs cuando se pasa entre conglomerados una referencia de interfaz de ingeniería;
- es notificada por un gestor de conglomerado cuando todas las copias de una referencia de interfaz de ingeniería se han suprimido en su conglomerado;
- detecta cuándo no se mantiene ninguna copia de referencia de interfaz de ingeniería fuera del conglomerado que soporta la interfaz de ingeniería a que se hace referencia, y notifica al correspondiente gestor de conglomerado;
- notifica a los gestores de las cápsulas en que hay poseedores de referencias de interfaz de ingeniería sobre una interfaz que haya fallado o haya sido suprimida.

NOTA – La notificación de eventos de rastreo de referencia de interfaz de ingeniería puede hacerse por medio de la función de notificación de evento.

La función de gestión de referencias de interfaz de ingeniería está constreñida por la política de gestión de referencias de interfaz de ingeniería de los dominios de gestión de referencias de interfaz de ingeniería a que se aplica.

14 Funciones de depositario

14.1 Función de almacenamiento

La función de almacenamiento almacena datos.

14.1.1 Conceptos

14.1.1.1 Depositario de datos: Objeto que proporciona la función de almacenamiento.

14.1.1.2 Interfaz contenedora: Interfaz de un depositario de datos que permite el acceso a datos.

14.1.2 Reglas

Un depositario de datos almacena conjuntos de datos. Cada conjunto de datos está asociado con una interfaz contenedora creada cuando se almacenan los datos. Una interfaz contenedora proporciona funciones para:

- obtener una copia de los datos almacenados por la interfaz;
- modificar los datos asociados con la interfaz;
- suprimir la interfaz contenedora y sus datos asociados.

NOTA – Los objetos incorporan acciones y estado (esto es, proceso y datos) y son por naturaleza persistentes. La función de punto de comprobación y recuperación o la función de replicación pueden utilizarse en una infraestructura de la función de almacenamiento para proporcionar estabilidad. Las funciones pueden utilizar otras instancias de la función de almacenamiento como un depositario para puntos de comprobación de conglomerado.

14.2 Función de organización de información

La función de organización de información gestiona un depositario de información descrito por un esquema de información e incluye todos o algunos de los elementos siguientes:

- modificación y actualización del esquema de información;
- interrogación del depositario mediante un lenguaje de interrogación;
- modificación y actualización del depositario.

La forma y la naturaleza de las interrogaciones y de las respuestas a las interrogaciones dependen del lenguaje de interrogación. La función de organización de información no permite modificaciones ni actualizaciones del depositario de información que son inconsecuentes con su esquema.

La función de organización de información puede modelarse como un depositario de objetos (con interfaces computacionales) que está en correspondencia con entidades y relaciones en un sistema ODP. Estos objetos soportan operaciones para:

- definir atributos, propiedades y relaciones para objetos en el depositario;
- añadir y suprimir objetos en el depositario;
- afirmar la certeza de, y suprimir, atributos, propiedades y relaciones para objetos seleccionados en el depositario;
- seleccionar objetos que satisfacen un predicado (es decir, un tipo) especificado en términos de atributos, propiedades y relaciones.

NOTAS

1 La función de organización de información pudiera derivar relaciones adicionales a las que son instanciadas directamente.

2 La función de organización de información puede utilizarse para mantener la relación entre un conglomerado y su punto de comprobación (esto es, una interfaz de almacenamiento en el que se puede crear o ganar acceso al punto de comprobación).

3 Para permitir la recuperación en respuesta a interacciones con un conglomerado que ha fallado, puede utilizarse la función de organización de información para mantener una relación entre un conglomerado y la interfaz en la que puede pedirse la recuperación del conglomerado.

4 Para permitir la reactivación en respuesta a interacciones con un conglomerado desactivado, puede utilizarse la función de organización de información para mantener una relación entre un conglomerado desactivado y la interfaz en la que puede pedirse la reactivación del conglomerado.

5 La función de organización de información permite mantener la información que necesita un reubicador para validar o actualizar una referencia de interfaz.

14.3 Función de reubicación

La función de reubicación gestiona un depositario de ubicaciones para interfaces, incluidas ubicaciones de funciones de gestión para el conglomerado que soporta esas interfaces.

14.3.1 Conceptos

14.3.1.1 Reubicador: Un objeto que proporciona la función de reubicación.

14.3.2 Reglas

Un reubicador tiene un directorio de ubicaciones para interfaces cuyas ubicaciones han sufrido cambios como consecuencia de actividades de gestión de dominio de comunicaciones (por ejemplo, el cambio de la dirección de red de un nodo) o de actividades de gestión de conglomerados tales como desactivación, traslado, replicación o recuperación de un punto de comprobación de conglomerado.

Se puede asociar una interfaz con un reubicador; los reubicadores pueden ser específicos de una sola interfaz o comunes a varias interfaces. Cuando el alcance de un reubicador abarca más de un dominio de gestión de referencias de interfaz de ingeniería, el medio por el cual se gana acceso al reubicador debe manifestar claramente el dominio de gestión de referencias de interfaz aplicable.

Cuando un reubicador está asociado con una interfaz, las actividades que cambian las ubicaciones de interfaces deben notificar al reubicador la nueva ubicación, en particular un gestor de conglomerado debe notificar al reubicador, para cada interfaz de cada objeto en el conglomerado, cuando el conglomerado es reubicado. Sólo es necesario registrar las interfaces de objetos cuyas ubicaciones hayan sufrido cambios. Cuando una referencia de interfaz de ingeniería deba mantenerse válida aunque el objeto que la soporta esté en un conglomerado desactivado, o en un conglomerado que ha fallado pero que había sido verificado anteriormente por punto de comprobación, el reubicador tiene que incorporar una política para la utilización de las funciones de coordinación para restaurar el conglomerado, por reactivación o recuperación, según proceda, cuando otro objeto trate de validar la referencia.

Un reubicador soporta interacciones que:

- registran un cambio en la ubicación de una interfaz identificada por una referencia de interfaz de ingeniería;
- valida la ubicación de una interfaz identificada por una referencia de interfaz de ingeniería (incluida, si es necesaria, la restauración del conglomerado que contiene el objeto que soporta la interfaz);
- establece la política para interactuar con funciones de coordinación (por ejemplo, la función de desactivación y reactivación) cuando se valida la ubicación de una interfaz identificada por una referencia de interfaz de ingeniería;

NOTA – Un objeto que efectúa la validación de una referencia de interfaz de ingeniería puede retener los resultados de la validación para optimizar el uso futuro de la referencia.

14.4 Función de depositario de tipos

La función de depositario de tipos gestiona un depositario de especificaciones de tipo y de relaciones de tipo. Tiene una interfaz para cada especificación de tipo que gestiona.

14.4.1 Reglas

La función de depositario de tipos puede ser informada de relaciones de tipos además de los que puede obtener por la comparación de las especificaciones de tipo. Un depositario de tipos no permitirá que se establezcan relaciones inconsecuentes.

Las especificaciones de tipo son inmutables.

La función de depositario de tipos incluye la creación de tipos y de sus interfaces asociadas.

Una interfaz de depositario de tipos para un tipo específico proporciona funciones para:

- interrogar la especificación del tipo;
- afirmar la certeza de relaciones entre el tipo en cuestión y otros tipos;
- interrogar relaciones entre el tipo en cuestión y otros tipos.

NOTA – Las relaciones de subtipificación para tipos de firma computacional se definen por las reglas de subtipificación indicadas en 7.2.4. No es necesario que un depositario de tipos habilite las relaciones de subtipos de firmas que habrán de computarse. Cuando en un depositario de tipos estén incluidos tipos de firmas, no se permitirá al depositario que imponga reglas o aserciones adicionales de subtipificación de firmas que contradigan las disposiciones de 7.2.4.

14.5 Función de comercio

La función de comercio media entre el anuncio y el descubrimiento de interfaces.

14.5.1 Conceptos

14.5.1.1 Oferta de servicio: Información sobre una interfaz, que incluye un identificador de la interfaz y su tipo de firma de interfaz computacional.

NOTAS

- 1 El identificador permite la vinculación a la interfaz.
- 2 La firma computacional permite a un comerciante («trader») asegurar que la importación de servicios selecciona ofertas de servicio que interactuarán de la manera prevista por el objeto importador.
- 3 En la oferta de servicio puede estilizarse información adicional para proporcionar una discriminación mayor que la incorporada en firmas de interfaz.

14.5.1.2 Exportación de servicio: Interacción con la función de comercio en la que se anuncia una oferta de servicio añadiéndola a un conjunto identificado de ofertas de servicio.

14.5.1.3 Importación de servicio: Interacción con la función de comercio en la que se busca en un conjunto identificado de ofertas de servicio para descubrir interfaces en el que está disponible un servicio que satisface un tipo especificado.

14.5.2 Reglas

La función de comercio proporciona importación de servicio y exportación de servicio, y su comportamiento se rige por una política de comercio que establece reglas sobre la manera de relacionar los conjuntos identificados en la exportación de servicio con los conjuntos identificados en la importación de servicio. En la importación de servicio, una función de comercio tiene que seleccionar solamente ofertas que satisfagan la política de la propia función de comercio, la política del exportador de la oferta de servicio y la política del importador de la oferta de servicio. La importación de servicio comprende la comprobación del subtipo/supertipo de firma de interfaz computacional. Puede comprender asimismo ulteriores niveles de comprobación, incluidas comprobaciones sobre capacidades de comportamiento y constricciones del entorno.

15 Funciones de seguridad

15.1 Conceptos

Los siguientes conceptos son comunes a todas las funciones de seguridad.

15.1.1 Política de seguridad: Conjunto de reglas que constriñe uno o más conjuntos de objetos.

15.1.2 Autoridad de seguridad: Administrador responsable de la implementación de una política de seguridad.

15.1.3 Dominio de seguridad: Dominio cuyos miembros están obligados a seguir una política de seguridad establecida y administrada por una autoridad de seguridad.

NOTA – La autoridad de seguridad es el objeto controlador para el dominio de seguridad.

15.1.4 Política de interacción de seguridad: Aspectos de las políticas de seguridad de diferentes dominios de seguridad que son necesarios para que se produzcan interacciones entre esos dominios.

15.2 Función de control de acceso

La función de control de acceso impide las interacciones no autorizadas con un objeto. Incluye una **función de decisión de control de acceso** y una **función de ejecución de control de acceso**. En el contexto del control de acceso, los objetos desempeñan, sea el papel de **iniciador**, sea el papel de **destino**. La función requiere una **información de control de acceso** sobre el destino, el iniciador y la interacción.

El iniciador pide que la función de control de acceso realice una interacción con el destino. Sobre la base de la información de control de acceso, la función de decisión de control de acceso decide si se permitirá o se denegará dicha interacción y, si se permite, la decisión será ejecutada por la función de ejecución de control de acceso.

NOTA – La función de decisión de control de acceso y la función de ejecución de control de acceso pueden ser proporcionadas por el objeto que desempeña el papel de destino, o por otros objetos.

15.3 Función de auditoría de seguridad

La función de auditoría de seguridad proporciona la supervisión y recogida de información sobre acciones relacionadas con la seguridad, y el subsiguiente análisis de la información para revisar las políticas, controles y procedimientos de seguridad.

La función de auditoría de seguridad incluye los elementos siguientes:

- **función de colector de alarma;**
- **función de examinador de alarma;**
- **función de analizador de pista de auditoría;**
- **función de archivador de pista de auditoría;**
- **función de registrador de auditoría;**
- **función de examinador de pista de auditoría;**
- **función de colector de pista de auditoría.**

15.4 Función de autenticación

La función de autenticación da la seguridad de la identidad pretendida de un objeto. En el contexto de autenticación, los objetos desempeñan uno de los siguientes papeles:

- **principal;**
- **reclamante** (sinónimos: demandante, declarante);
- **tercero de confianza.**

La autenticación requiere la utilización de **información de autenticación de intercambio**.

NOTAS

1 Cualquier objeto identificable en un sistema ODP puede ser el principal para la autenticación, lo que incluye tanto los objetos que modelan personas como los que modelan sistemas de computadores.

2 El objeto que inicia una autenticación no es necesariamente el reclamante.

Hay dos formas de autenticación:

- **autenticación de entidad par**, que corrobora la identidad de un principal en el contexto de una relación de comunicación;
- **autenticación del origen de datos**, que corrobora la identidad del principal responsable de una determinada unidad de datos.

NOTA – En la Rec. UIT-T X.811 | ISO/CEI 10181-2 se indican las categorías de mecanismo de autenticación.

En una autenticación que abarca dos objetos, cualquiera de los dos objetos puede desempeñar el papel de reclamante. Cuando ambos objetos desempeñan el papel de reclamante, el estilo de autenticación se conoce por **autenticación mutua**. La información de autenticación de intercambio se pasa del objeto iniciador al objeto respondedor y ulteriores informaciones de autenticación de intercambio pueden pasarse en el sentido opuesto. Puede haber aún otros intercambios: diferentes mecanismos de autenticación requieren diferentes números de intercambios. La autenticación de entidad par implica siempre la interacción con el reclamante. La autenticación del origen de datos no implica la interacción con el reclamante.

Un reclamante soporta operaciones para adquirir la información necesaria para una instancia de autenticación y para generar información de autenticación de intercambio. Un verificador soporta operaciones para adquirir la información necesaria para una instancia de autenticación y para la verificación de la información de autenticación de intercambio recibida y/o para generarla. La información puede intercambiarse con un servidor de autenticación y, o bien con el reclamante solamente, o con el verificador solamente, o con el reclamante y el verificador, sea antes o en el curso de intercambios de autenticación.

La función de autenticación puede utilizar la función de gestión de claves.

15.5 Función de integridad

La función de integridad detecta, o impide, o detecta e impide la creación, modificación o supresión no autorizadas de datos.

La función de integridad incluye las funciones siguientes:

- **protección;**
- **validación;**
- **desprotección.**

En el contexto de integridad, los objetos desempeñan uno o más de los siguientes papeles:

- **originador de datos protegidos en integridad;**
- **recibiente de datos protegidos en integridad.**

Los datos protegidos en integridad se pasan del originador al recibiente. Un originador de datos protegidos en integridad soporta una interfaz que proporciona la función de protección. Un recibiente de datos protegidos en integridad soporta una interfaz que proporciona las funciones de validación o desprotección.

La función de integridad puede utilizar la función de gestión de claves.

15.6 Función de confidencialidad

La función de confidencialidad impide la revelación no autorizada de información.

La función de confidencialidad incluye las funciones de **ocultación** y **revelación**.

En el contexto de confidencialidad, los objetos desempeñan uno o más de los papeles siguientes:

- **originador de información protegida en confidencialidad;**
- **recibiente de información protegida en confidencialidad.**

La información protegida en confidencialidad se pasa del originador al recipiente. Un originador de información protegida en confidencialidad soporta una interfaz que proporciona la función de ocultación. Un recipiente de información protegida en confidencialidad soporta una interfaz que proporciona la función de revelación.

La función de confidencialidad puede utilizar la función de gestión de claves.

15.7 Función de no repudio

La función de no repudio impide que un objeto que ha intervenido en una interacción niegue haber participado en toda o parte de la interacción.

En el contexto de no repudio, los objetos desempeñan uno de los papeles siguientes:

- **originador (de datos no repudiables);**
- **recibiente (de datos no repudiables);**
- **generador de prueba;**
- **usuario de prueba;**
- **verificador de prueba;**
- **solicitante del servicio de no repudio;**
- **notario;**
- **adjudicador.**

La función de no repudio utiliza prueba de no repudio. En el caso de no repudio con prueba del origen, el originador desempeña el papel de generador de prueba de no repudio para la interacción de originación e incluye esta prueba en un reconocimiento de participación (dícese también acuse de participación) en la interacción. El recipiente desempeña el papel de usuario de prueba y utiliza los servicios de un verificador de prueba (que puede ser él mismo) para adquirir confianza en la idoneidad de la prueba. En el caso de no repudio con prueba de entrega, el recipiente desempeña el papel de generador de prueba de no repudio para la interacción de entrega e incluye su prueba en un acuse de participación en la interacción. El originador desempeña el papel de usuario de prueba y utiliza los servicios de un verificador de prueba (que puede ser él mismo) para adquirir confianza en la idoneidad de la prueba.

Un notario proporciona funciones requeridas por el originador, por el recipiente, o por ambos. Entre estas funciones pueden estar las de notarización, indicación de tiempo, supervisión, certificación, generación de certificado, generación de firma, verificación de firma, y entrega, especificadas en la Rec. UIT-T X.813 | ISO/CEI 10181-4.

En caso de disputa, un adjudicador recoge informaciones y pruebas de las partes en discordia (y, facultativamente, del notario) y aplica una función de resolución como se indica en la mencionada Rec. UIT-T X.813 | ISO/CEI 10181-4.

La función de no repudio puede utilizar la función de gestión de claves.

15.8 Función de gestión de claves

La función de gestión de claves proporciona medios para la gestión de claves criptográficas e incluye los elementos siguientes:

- **generación de claves;**
- **registro de claves;**
- **certificación de claves;**
- **desregistro de claves;**
- **distribución de claves;**

- **almacenamiento de claves;**
- **archivo de claves;**
- **supresión de claves.**

Dentro del contexto de gestión de claves, los objetos pueden desempeñar uno o más de los siguientes papeles:

- **autoridad de certificación;**
- **centro de distribución de claves;**
- **centro de traducción de claves.**

Una autoridad de certificación es un tercero de confianza que crea y asigna certificados como se especifica en la ISO/CEI 11770-1. Un centro de distribución de claves proporciona medios para establecer, en forma securizada, información de gestión de claves entre objetos autorizados para obtenerla. Un centro de traducción de claves es una forma específica de centro de distribución de claves, que establece información de gestión de claves entre objetos en diferentes dominios de seguridad.

16 **Transparencia de distribución ODP**

La transparencia de distribución es selectiva en los sistemas ODP. Este modelo de referencia describe cómo alcanzar las siguientes transparencias de distribución:

- transparencia de acceso;
- transparencia de fallo;
- transparencia de ubicación;
- transparencia de traslado;
- transparencia de persistencia;
- transparencia de reubicación;
- transparencia de replicación;
- transparencia de transacción.

Las normas ODP pueden definir:

- refinamientos de las descripciones en este modelo de referencia, y
- transparencias de distribución adicionales requeridas para esas normas.

Las transparencias se definen como constricciones a la correspondencia, de una especificación computacional que contiene un esquema de transparencia, a una especificación que utiliza funciones y estructuras de ingeniería ODP específicas para proporcionar la forma de enmascaramiento requerida.

El comportamiento de stubs, vinculadores, objetos de protocolo e interceptores en canales viene determinado por la combinación de transparencias de distribución (por ejemplo, transparencia de acceso, transparencia de reubicación) que se aplica al canal.

En algunas normas implementables que requieren más de una transparencia de distribución, pudiera especificarse una composición de reglas para objetos que soportan transparencias individuales. En otras normas implementables que requieren más de una transparencia de distribución, un solo objeto podría proporcionar la transparencia de distribución combinada.

Las descripciones de transparencia en este modelo de referencia soportan por lo menos las siguientes combinaciones de transparencias de distribución:

- a) transparencia de acceso y de ubicación;
- b) transparencia de a) y de reubicación;
- c) transparencia de b) y de traslado;
- d) transparencia de b) y de recurso;
- e) transparencia de b) y de fallo;
- f) transparencia de a) y de transacción;
- g) transparencia de b) y de transacción.

16.1 Transparencia de acceso

La transparencia de acceso enmascara diferencias en la representación de datos y en los mecanismos de invocación para permitir el interfuncionamiento de objetos.

La transparencia de acceso se obtiene mediante la selección de una estructura de canal adecuada (por ejemplo, una estructura en la cual los stubs proporcionen conversiones apropiadas, por ejemplo organizando los datos en su forma de representación canónica).

16.2 Transparencia de fallo

La transparencia de fallo enmascara, ocultándoselos a un objeto, el fallo y la posible recuperación de otros objetos, o los suyos propios, para proporcionar una tolerancia a las averías (o faltas).

16.2.1 Conceptos

16.2.1.1 Esquema de estabilidad: Especificación de modos de fallo que un objeto no presentará.

16.2.2 Reglas

El refinamiento de transparencia de fallo puede satisfacer un esquema de estabilidad por uno de los métodos siguientes:

- situando el objeto en un nodo cuya infraestructura excluya los fallos especificados;
- utilizando la función de punto de comprobación y recuperación para hacer que el objeto sea estable;
- utilizando la función de replicación para hacer que el objeto sea estable.

16.2.2.1 Replicación

En el caso de replicación, el refinamiento de transparencia de fallo incluye los pasos siguientes:

- definición de una interfaz de gestión de objeto que soporte la verificación por punto de comprobación y la supresión del objeto en el caso del objeto computacional;
- introducción de una función de replicación;
- establecimiento de una política de replicación para los conglomerados que contienen el objeto;
- asociación de un reubicador que soporte la replicación con cada una de las interfaces del objeto.

En la arquitectura definida por este modelo de referencia, la transparencia de fallo basada en la replicación de un conglomerado requiere la transparencia de reubicación.

16.2.2.2 Punto de comprobación y recuperación

En el caso de punto de comprobación y recuperación, el refinamiento de transparencia de fallo incluye los pasos siguientes:

- definición de una interfaz de gestión de objeto que soporte la verificación por punto de comprobación y la supresión del objeto en el caso del objeto computacional;
- introducción de una función de punto de comprobación y recuperación;
- establecimiento de una política de punto de comprobación y recuperación para los conglomerados que contienen el objeto;
- asociación de un reubicador que soporte la replicación con cada una de las interfaces del objeto.

La transparencia de fallo basada en la verificación por punto de comprobación y la recuperación de un conglomerado requiere la transparencia de reubicación.

16.3 Transparencia de ubicación

La transparencia de ubicación enmascara la utilización de información sobre la ubicación en el espacio cuando se efectúa la identificación y la vinculación a interfaces. Permite a los objetos ganar acceso a las interfaces sin utilizar información de ubicación.

16.4 Transparencia de traslado

La transparencia de traslado enmascara, ocultándosela a un objeto, la aptitud de un sistema para cambiar la ubicación de ese objeto.

16.4.1 Conceptos

16.4.1.1 Esquema de movilidad: Especificación que impone constricciones a la movilidad de un objeto.

Un esquema de movilidad incluye:

- constricciones de latencia sobre las interacciones con el objeto;
- constricciones de rendimiento sobre los hilos del objeto;
- constricciones de seguridad sobre la ubicación del objeto.

16.4.2 Reglas

La transparencia de traslado se proporciona mediante el empleo de la función de traslado para coordinar la ubicación de un objeto con el fin de satisfacer un esquema de movilidad. El refinamiento de transparencia de traslado incluye los siguientes pasos:

- definición de una interfaz de gestión de objeto que soporte la verificación por punto de comprobación y la supresión del objeto en el caso del objeto computacional;
- introducción de una función de traslado;
- establecimiento de una política de traslado para los conglomerados que contienen el objeto;
- asociación de un reubicador con cada una de las interfaces del objeto.

En la arquitectura definida por este modelo de referencia, la transparencia de traslado requiere la transparencia de reubicación para todos los canales vinculados al conglomerado.

16.5 Transparencia de persistencia

La transparencia de persistencia enmascara, ocultándoselas a un objeto, la desactivación y reactivación de otros objetos (o las suyas propias).

16.5.1 Conceptos

16.5.1.1 Esquema de persistencia: Especificación de constricciones sobre la utilización de funciones específicas de procesamiento, almacenamiento y comunicación.

16.5.2 Reglas

La transparencia de persistencia se proporciona mediante el empleo de la función de reactivación y desactivación para coordinar la desactivación y reactivación de conglomerados con el fin de satisfacer un esquema de persistencia. El refinamiento de transparencia de persistencia incluye los siguientes pasos:

- definición de una interfaz de gestión de objeto que soporte la verificación por punto de comprobación y la supresión del objeto en el caso del objeto computacional;
- introducción de una función de desactivación y reactivación;
- establecimiento de una política de desactivación y reactivación para los conglomerados que contienen el objeto;
- asociación de un reubicador que soporte la reactivación con cada una de las interfaces del objeto.

En la arquitectura definida por este modelo de referencia, la transparencia de persistencia requiere la transparencia de reubicación para todos los canales vinculados al conglomerado.

16.6 Transparencia de reubicación

La transparencia de reubicación enmascara la reubicación de una interfaz, ocultándosela a otras interfaces vinculadas con ella.

Las transparencia de reubicación requiere:

- La asociación de un reubicador a cada interfaz en el conglomerado.
- El envío, a los reubicadores, de información sobre los cambios de ubicación del objeto.
- Vinculadores para intercambiar datos adicionales en interacciones a través de un canal para confirmar la validez de la vinculación soportada por el canal. Estos datos se derivan de la ubicación, en el espacio y en el tiempo, asociada con las referencias de interfaz de ingeniería de las interfaces vinculadas al canal.

Cuando un vinculador detecta que un canal ha sido invalidado por la reubicación de un objeto (por ejemplo, al observar un fallo de comunicación), el vinculador tiene que validar las referencias de interfaz de ingeniería para las interfaces con las que estaba vinculado el canal y, si es necesario, restablecer el canal. La validación la efectúa la función de reubicación.

NOTA – Si el canal es invalidado por la desactivación de un objeto, o por el fallo de un objeto anteriormente verificado por punto de comprobación, el reubicador incorporará un procedimiento para restaurar el conglomerado que contiene el objeto, como parte de la validación.

16.7 Transparencia de replicación

La transparencia de replicación enmascara la utilización de un grupo de objetos mutuamente compatibles en comportamiento para soportar una interfaz.

16.7.1 Conceptos

16.7.1.1 Esquema de replicación: Especificación de constricciones sobre la replicación de un objeto que incluye constricciones sobre la disponibilidad del objeto y sobre el rendimiento del objeto.

16.7.2 Reglas

La transparencia de replicación se proporciona mediante el empleo de la función de replicación para coordinar la replicación de un objeto con el fin de satisfacer un esquema de replicación. El refinamiento de transparencia de replicación incluye los siguientes pasos:

- definición de una interfaz de gestión de objeto que soporte la verificación por punto de comprobación y la supresión del objeto en el caso del objeto computacional;
- introducción de una función de replicación;
- establecimiento de una política de replicación para los conglomerados que contienen el objeto;
- asociación de un reubicador con cada una de las interfaces del objeto.

En la arquitectura definida por este modelo de referencia, la transparencia de replicación requiere la transparencia de reubicación para el conglomerado.

16.8 Transparencia de transacción

La transparencia de transacción enmascara la coordinación de las actividades de los objetos que constituyen una configuración de objetos, para lograr la consistencia.

16.8.1 Concepto

16.8.1.1 Esquema de transacción: Esquema dinámico y esquema invariante que definen transacciones y sus dependencias.

16.8.2 Reglas

La transparencia de transacción se proporciona mediante el empleo de la función de transacción para coordinar el comportamiento de un objeto con el fin de satisfacer un esquema de transacción. El refinamiento de transferencia de transacción incluye los siguientes pasos:

- derivación, a partir del esquema transaccional, de las políticas relativas a la función de transacción;
- adición de operaciones de punto de comprobación y de recuperación para el estado del objeto;
- sustitución de vinculaciones entre los objetos mediante una función de transacción;
- ampliación de las interfaces de los objetos computacionales.

La ampliación de las interfaces de los objetos incluye funciones para notificar la realización de acciones de interés y para recuperar el estado del objeto tras una anulación.

Anexo A

Reglas formales para el establecimiento de supertipos/subtipos de interfaces computacionales

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se definen las reglas formales para el establecimiento de subtipos (brevemente, reglas de subtipificación) de firma de interfaces computacionales. Los tipos de firma de interfaces computacionales pueden ser de orden superior como se deduce de 7.2.2.4 reglas de parámetros. Este anexo sólo formaliza un subconjunto de primer orden de las reglas de subtipificación; la formalización de características de orden superior de tipos de firma de interfaces computacionales queda en estudio.

El presente anexo define un sistema de tipos de primer orden constituido por un lenguaje de tipo simple junto con reglas de igualdad de tipos y reglas de subtipificación de firmas. Describe también un algoritmo correcto y completo de comprobación de tipos, para el sistema de tipos. Mediante el lenguaje de tipo se definen tipos de firma de interfaz de señales, tipos de firma de interfaz de operaciones y tipos de firma de interfaz de trenes. Dado que la subtipificación de firma de interfaz de trenes se define sólo parcialmente en 7.2.4.2, este anexo sólo formaliza la regla de subtipificación que se aplica entre flujos correspondientes.

A.1 Notaciones y convenios

Se utilizan las siguientes notaciones:

- α, β, γ , etc., denotan tipos;
- t, s , etc., denotan identificadores para tipos (es decir, variables de tipo) y tipos de fundamento (es decir, constantes de tipo); el conjunto de variables tipo (y de constantes tipo) se designa por T_{var} ;
- a, b, c, a_1, a_2, a_n , etc., denotan identificadores o *etiquetas* para elementos de estructuras en el lenguaje de tipo; el conjunto de etiquetas se designa por Λ ;
- $\alpha[\beta/t]$ denota la sustitución de β por t en α ;
- *Nil* denota una constante de tipo predefinida.

A.2 Sistema de tipos

El sistema de tipos comprende constantes de tipo, funciones, productos cartesianos, registros («records»), definiciones recursivas de uniones rotuladas. El lenguaje de tipo, *Type*, se da por la gramática de la Figura A.1.

$a ::=$	t
	\perp
	\top
	$\alpha \rightarrow \beta$
	$\alpha_1 \times \dots \times \alpha_n$
	$\langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle$
	$[c_1 : \gamma_1, \dots, c_n : \gamma_n]$
	$\mu t. \alpha$

Figura A.1 – Sintaxis abstracta para declaraciones de tipo

Los tipos de fundamento («ground types») se designan por \top (superior) y \perp (inferior). Representan el elemento más grande y el elemento más pequeño, respectivamente, en la relación de subtipo. Las funciones se denotan así: $\alpha \rightarrow \beta$. Los productos cartesianos se denotan así: $\alpha_1 \times \dots \times \alpha_n$. Las uniones se denotan así: $[c_1 : \gamma_1, \dots, c_n : \gamma_n]$. Los registros («records») se denotan así: $\langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle$.

μ es un operador de vinculación de variable. Los tipos recursivos pueden construirse vinculando tipos a identificadores y referenciando un identificador para un tipo, en otro.

Se utilizan paréntesis para determinar la precedencia cuando sea necesario. En su ausencia, \rightarrow asocia hacia la derecha y el campo de acción de μ se extiende hacia la derecha lo más posible.

El conjunto de variables libres que aparecen en α se denotan así: $FV(\alpha)$.

A.2.1 Reglas de tipificación

En esta cláusula se indican reglas de igualdad de tipos y reglas de subtipificación para el lenguaje dado.

Un tipo α es *contractivo* en la variable de tipo t , denotado por $\alpha \downarrow t$, si, o bien t no aparece libre en α , o bien α puede reescribirse mediante un desarrollo como un tipo de una de las formas siguientes:

- $\alpha_1 \rightarrow \alpha_2$;
- $\langle a_1 : \alpha_1, \dots, a_m : \alpha_m \rangle$;
- $[c_1 : \gamma_1, \dots, c_n : \gamma_n]$;
- $\alpha_1 \times \dots \times \alpha_n$.

Las reglas de igualdad de tipos se indican en la Figura A.2. La igualdad de tipo se denota por $=$.

(E.1)	$\alpha = \alpha$
(E.2)	$\alpha = \beta \Rightarrow \beta = \alpha$
(E.3)	$\alpha = \beta, \beta = \gamma \Rightarrow \alpha = \gamma$
(E.4)	$\alpha_1 = \alpha_2, \beta_1 = \beta_2 \Rightarrow \alpha_1 \rightarrow \beta_1 = \alpha_2 \rightarrow \beta_2$
(E.5)	$\alpha = \beta \Rightarrow \mu t. \alpha = \mu t. \beta$
(E.6)	$\forall i \in \{1, \dots, n\}, \alpha_i = \beta_i \Rightarrow \alpha_1 \times \dots \times \alpha_n = \beta_1 \times \dots \times \beta_n$
(E.7)	$\forall i \in \{1, \dots, n\}, \alpha_i = \beta_i \Rightarrow \langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle = \langle a_1 : \beta_1, \dots, a_n : \beta_n \rangle$
(E.8)	$\forall i \in \{1, \dots, n\}, \alpha_i = \beta_i \Rightarrow [a_1 : \alpha_1, \dots, a_n : \alpha_n] = [a_1 : \beta_1, \dots, a_n : \beta_n]$
(E.9)	$\mu t. t = \perp$
(E.10)	$\alpha[\mu t. \alpha / t] = \mu t. \alpha$
(E.11)	$\alpha[\beta / t] = \beta_1, \alpha[\beta_2 / t] = \beta_2 \Rightarrow \alpha \downarrow t \Rightarrow \beta_1 = \beta_2$

Figura A.2 – Reglas de igualdad de tipos

Las reglas de subtipificación se dan en forma de reglas de inferencia sobre juicios semejantes a los del lenguaje Prolog. Los juicios son de la forma $\Gamma \vdash \alpha \leq \beta$, donde Γ es un conjunto de supuestos de subtipificación sobre variables de tipo de la forma $\{t_1 \leq s_1, \dots, t_n \leq s_n\}$. Una regla típica puede tener la forma siguiente:

$$\Gamma \vdash \alpha_1 \leq \beta_1, \Gamma \vdash \alpha_2 \leq \beta_2 \Rightarrow \Gamma \vdash \alpha \leq \beta$$

Informalmente, esto significa que para determinar si $\Gamma \vdash \alpha \leq \beta$ es válida, hay que tratar primero de determinar si $\Gamma \vdash \alpha_1 \leq \beta_1$ y $\Gamma \vdash \alpha_2 \leq \beta_2$. Si se cumplen estas dos submetas se puede llegar a la conclusión de que α es un subtipo de β .

Las reglas de subtipificación se indican en la Figura A.3. Puede decirse que α es un subtipo de β si $\emptyset \vdash \alpha \leq \beta$ puede derivarse utilizando las reglas de subtipificación y las reglas de igualdad.

A.2.2 Definiciones de tipos

Los elementos de *Type* se definen por conjuntos de ecuaciones mutuamente dependientes, modelados como entornos bien formados. Un entorno es una correspondencia finita entre variables de tipo y tipos pertenecientes a T , donde T es un subconjunto no recursivo de *Type*. Un entorno bien formado Υ es un entorno tal que todas las variables libres de un tipo α asociadas con una variable t en el dominio de Υ pertenecen al dominio de Υ . Intuitivamente, cada variable de tipo en un entorno representa un tipo. Las asociaciones entre variables de tipo y elementos de T en un entorno pueden considerarse como ecuaciones de definición mutuamente dependientes para los tipos correspondientes.

(S.1)	$\alpha = \beta \Rightarrow \Gamma \vdash \alpha \leq \beta$
(S.2)	$\Gamma \vdash \alpha \leq \beta, \Gamma \vdash \beta \leq \gamma \Rightarrow \Gamma \vdash \alpha \leq \gamma$
(S.3)	$t \leq s \in \Gamma \Rightarrow \Gamma \vdash t \leq s$
(S.4)	$\Gamma \vdash \perp \leq \alpha$
(S.5)	$\Gamma \vdash \alpha \leq \top$
(S.6)	$\Gamma \vdash \alpha_2 \leq \alpha_1, \Gamma \vdash \beta_1 \leq \beta_2 \Rightarrow \Gamma \vdash \alpha_1 \rightarrow \beta_1 \leq \alpha_2 \rightarrow \beta_2 \leq$
(S.7)	$\forall i \in \{1, \dots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash \langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle \leq \langle a_1 : \beta_1, \dots, a_n : \beta_n \rangle$ con $n \leq m$
(S.8)	$\forall i \in \{1, \dots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash [a_1 : \alpha_1, \dots, a_n : \alpha_n] \leq [a_1 : \beta_1, \dots, a_n : \beta_n]$ con $n \leq m$
(S.9)	$\forall i \in \{1, \dots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash \alpha_1 \times \dots \times \alpha_n \leq \beta_1 \times \dots \times \beta_n$
(S.10)	$\Gamma \cup \{t \leq s\} \vdash \alpha \leq \beta \Rightarrow \Gamma \vdash \mu t. \alpha \leq \mu s. \beta$ con t sólo en α ; s sólo en β , t, s no en Γ

Figura A.3 – Reglas de subtificación

Formalmente, sea Υ_{wf} el conjunto de entornos bien formados, y defínase $f : A \rightarrow_f B$ como una función parcial de A a B con un dominio finito; $FV(\alpha)$ denota el conjunto de variables libres que aparecen en α :

$$\Upsilon_{wf} =_{def} \{ \Upsilon : T_{var} \rightarrow_f Type \mid \forall t, t' \in dom(\Upsilon), t' \in FV(\Upsilon(t)) \Rightarrow t' \in dom(\Upsilon) \}$$

Sea $\Upsilon = \{ t \mapsto \alpha, t_1 \mapsto \alpha_1, \dots, t_q \mapsto \alpha_q \}$. $\Upsilon \setminus t$ denota el siguiente entorno:

$$\Upsilon \setminus t =_{def} \{ t_1 \mapsto \alpha_1, \dots, t_q \mapsto \alpha_q \}$$

El tipo asociado con una variable de tipo t en el contexto de un entorno bien formado Υ ($t \in dom(\Upsilon)$) se define de modo que sea $Val(t, \Upsilon)$, donde Val es la función sobre tipos y entornos definida recursivamente en la Figura A.4. Así, todo elemento de $Type$ puede definirse como $Val(t, \Upsilon)$, donde Υ es un entorno bien formado y $t \in dom(\Upsilon)$.

(IT.1)	$Val(\perp, \Upsilon) = \perp$
(IT.2)	$Val(\top, \Upsilon) = \top$
(IT.3)	$Val(Nil, \Upsilon) = Nil$
(IT.4)	$Val(\alpha \rightarrow \beta, \Upsilon) = Val(\alpha, \Upsilon) \rightarrow Val(\beta, \Upsilon)$
(IT.5)	$Val(\langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle, \Upsilon) = \langle a_1 : Val(\alpha_1, \Upsilon), \dots, a_n : Val(\alpha_n, \Upsilon) \rangle$
(IT.6)	$Val([a_1 : \alpha_1, \dots, a_n : \alpha_n], \Upsilon) = [a_1 : Val(\alpha_1, \Upsilon), \dots, a_n : Val(\alpha_n, \Upsilon)]$
(IT.7)	$Val(\alpha_1 \times \dots \times \alpha_n, \Upsilon) = Val(\alpha_1, \Upsilon) \times \dots \times Val(\alpha_n, \Upsilon)$
(IT.8)	si $t \in dom(\Upsilon)$ entonces $Val(t, \Upsilon) = t$
(IT.10)	si $t \in dom(\Upsilon)$ entonces $Val(t, \Upsilon) = \mu t. Val(\Upsilon(t), \Upsilon \setminus t)$

Figura A.4 – Semántica de definiciones de tipos de interfaz

A.2.3 Un algoritmo para la comprobación de tipos

Esta subcláusula define un algoritmo para comprobación de tipos que es correcto y completo en lo que respecta a las reglas de igualdad de tipos y a las reglas de subtificación antes indicadas. El algoritmo comprende dos entornos bien formados ε_1 y ε_2 tales que $dom(\varepsilon_1) \cap dom(\varepsilon_2) = \emptyset$ (los tipos que se van a comparar se asocian a dos variables, una en ε_1 y la otra en ε_2). Se describe como un conjunto de reglas de inferencia que comprenden $\varepsilon =_{def} \varepsilon_1 \cup \varepsilon_2$ y un conjunto Σ de la forma $\{t_1 \leq s_1, \dots, t_n \leq s_n\}$ que registra la inclusión de variables descubiertas durante la ejecución del algoritmo. Una regla de inferencia corresponde a una implicación lógica de juicios de la forma $\Sigma, \varepsilon \vdash \alpha \leq \beta$. Un juicio capta intuitivamente que la aserción $\alpha \leq \beta$ es válida en el contexto de Σ y ε . Los juicios iniciales $\{t_1 \leq s_1, \dots, t_n \leq s_n\}$ tienen que ser tales que $\{t_1, s_1, \dots, t_n, s_n\} \cap dom(\varepsilon) = \emptyset$.

Las reglas de inferencia se dan en la Figura A.5. En la Figura A.5, $\alpha, \beta \in Type$, t, s denotan variables arbitrarias, u denota variables no en $dom(\varepsilon)$.

Dada una meta inicial $\Sigma, \varepsilon \vdash \alpha \leq \beta$, el algoritmo consiste en aplicar en sentido inverso las reglas de inferencia, generando submetas en casos (rec), (fun), (rcd), (pro) y (uni). Un árbol de metas construido de esta manera se llama un árbol de ejecución. Un árbol de ejecución es siempre finito. Si $t \leq s$ es un supuesto que ha de añadirse a Σ , entonces t y s son variables de tipo en $dom(\varepsilon)$; asimismo, las reglas (fun), (pro), (uni) y (rcd) reducen el tamaño de la meta actual al remplazarla por subexpresiones de la meta, y cada aplicación de (rec) agranda Σ .

(assmp)	$t \leq s \in \Sigma \Rightarrow \Sigma, \varepsilon \vdash t \leq s$
(bot)	$\Sigma, \varepsilon \vdash \perp \leq \beta$
(top)	$\Sigma, \varepsilon \vdash a \leq T$
(var)	$\Sigma, \varepsilon \vdash u \leq u$
(fun)	$\Sigma, \varepsilon \vdash \alpha_2 \leq \alpha_1, \Sigma, \varepsilon \vdash \beta_1 \leq \beta_2 \Rightarrow \Sigma, \varepsilon \vdash \alpha_1 \rightarrow \beta_1 \leq \alpha_2 \rightarrow \beta_2 \leq$
(rcd)	$\forall i \in \{1, \dots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash \langle a_1 : \alpha_1, \dots, a_n : \alpha_n \rangle \leq \langle a_1 : \beta_1, \dots, a_n : \beta_n \rangle$ con $n \leq m$
(uni)	$\forall i \in \{1, \dots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash [a_1 : \alpha_1, \dots, a_n : \alpha_n] \leq \{a_1 : \beta_1, \dots, a_m : \beta_m\}$ con $n \leq m$
(pro)	$\forall i \in \{1, \dots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash \alpha_1 \times \dots \times \alpha_n \leq \beta_1 \times \dots \times \beta_n$
(rec)	$\Sigma \cup \{t \leq s\}, \varepsilon \vdash \varepsilon(t) \leq \varepsilon(s) \Rightarrow \Sigma, \varepsilon \vdash t \leq s$

Figura A.5 – Reglas de interferencia de comprobación de tipo

Un árbol de ejecución *tiene éxito* si todas las hojas corresponden a una aplicación de una de las reglas (assmp), (bot), (top) o (var). Un árbol de ejecución *fracasa* si al menos una hoja es una meta incumplida (es decir, si no se le puede aplicar ninguna regla). Si el árbol de ejecución correspondiente a la meta $\emptyset \varepsilon \vdash \alpha \leq \beta$ tiene éxito, esto se denota por $\vdash_A \alpha \leq \beta$.

Dados tipos recursivos α y β , tales que $\alpha = Val(t_1, \varepsilon_1)$ y $\beta = Val(t_2, \varepsilon_2)$ (siendo ε_1 y ε_2 como se ha indicado antes), el algoritmo induce una relación \leq_A por la siguiente definición:

$$\alpha \leq_A \beta \Leftrightarrow \vdash_A t_1 \leq t_2$$

La nueva relación de subtipificación coincide con la anterior, es decir, el algoritmo es correcto y completo con respecto a las reglas de igualdad de tipos y a la regla de subtipificación:

- dados α, β en T , si $\alpha \leq_A \beta$ entonces $\alpha \leq_R \beta$;
- dados α, β en T , si $\alpha \leq_R \beta$ entonces $\alpha \leq_A \beta$.

A.3 Tipos de firma de interfaz de señales

Los tipos de firma de interfaz de señales son formalizados interpretándolos en el lenguaje *Type*. El conjunto de tipos de firma de interfaz de señales se designa por $Type_{sig}$. Los elementos de $Type_{sig}$ se definen abstractamente mediante el empleo de dos funciones: $intype: Type_{sig} \rightarrow Type$ y $outype: Type_{sig} \rightarrow Type$. En un tipo dado de firma de interfaz de señales, *intype* describe el conjunto de señales iniciadoras, y *outype* describe el conjunto de señales respondedoras.

Los elementos de *Type* asociados con un tipo de firma de interfaz de señales mediante las funciones *intype* y *outype* se definen por entornos bien formados en el mismo dominio que el subconjunto de *Type* definido por la gramática de la Figura A.6, donde se supone que las etiquetas $a_i \in \{1, \dots, q\}$ son distintas. En efecto, la Figura A.6 proporciona una sintaxis abstracta para firmas de interfaz de señales. Las etiquetas a_i corresponden a nombres de señal. Las producciones *Arg* corresponden a parámetros de señales. Las producciones *Sigsig* corresponden a firmas de señales individuales. La forma funcional adoptada para la firma de señal individual destaca la analogía con firmas de anuncios.

La relación de subtipificación sobre tipos de interfaz de señales, \leq_s , se define por:

$$\forall \iota_1, \iota_2 \in Type_{sig}, \iota_1 \leq \iota_2 \equiv \iota_1.intype \leq \iota_2.intype \wedge \iota_2.outype \leq \iota_1.outype$$

α	::=	$\langle a_i : Sigsig, \dots, a_q : Sigsig \rangle$
$Sigsig$::=	$Arg \rightarrow Nil$
Arg	::=	$Nil \mid t_1 \times \dots \times t_p$

Figura A.6 – Sintaxis abstracta para tipos de firma de interfaz de señales

A.4 Tipos de firma de interfaz de operaciones

Los tipos de firma de interfaz de *servidor* operacional se formalizan interpretándolas en el lenguaje *Type* (los tipos de firma de interfaz de *cliente* de operaciones pueden derivarse inmediatamente por complementación). El conjunto de tipos de interfaz de servidor operacional se denota por $Type_o(S)$. Los elementos de $Type_o(S)$ se definen abstractamente mediante el empleo de la función $optype: Type_o(S) \rightarrow Type$.

Los elementos de *Type* asociados con un tipo de firma de interfaz de operaciones mediante la función *optype* se definen por entornos bien formados en el mismo dominio que el subconjunto de *Type* definido por la gramática de la Figura A.7, donde se supone que las etiquetas $a_i, i \in \{1, \dots, q\}$ son distintas, y donde se supone que las etiquetas $c_i, i \in \{1, \dots, q\}$ son distintas en el contexto de una producción *Opsig*.

α	::=	$\langle a_i : Opsig, \dots, a_q : Opsig \rangle$
<i>Opsig</i>	::=	$Arg \rightarrow Term \mid Arg \rightarrow Nil$
<i>Term</i>	::=	$[c_1 : Arg, \dots, c_q : Arg]$
<i>Arg</i>	::=	$Nil \mid t_1 \times \dots \times t_p$

Figura A.7 – Sintaxis abstracta para tipos de firma de interfaz de operaciones

En efecto, la Figura A.7 proporciona una sintaxis abstracta para firmas de interfaz de operaciones. Las producciones *Opsig* corresponden a firmas de operación individuales. Específicamente, las producciones *Opsig* de la forma $Arg \rightarrow Term$ en la Figura A.1 corresponden a interrogaciones. Las producciones *Opsig* de la forma $Arg \rightarrow Nil$ corresponden a anuncios. Las producciones *Arg* corresponden a parámetros en la invocación. Las producciones *Term* corresponden a terminaciones. *Nil* al lado izquierdo de una producción *Opsig* significa que la invocación dada no tiene parámetros. *Nil* al lado derecho de una producción *Opsig* (esto es, en una firma de anuncio) significa que no se espera una terminación. Las etiquetas a_i corresponden a nombres de operación. Las etiquetas c_1 corresponden a nombres de terminación.

La relación de subtipificación sobre tipos de interfaz operacional de servidor, \leq_o , se define por:

$$\forall \iota_1, \iota_2 \in Type_o(S), \iota_1 \leq \iota_2 \equiv \iota_1.optype \leq \iota_2.optype$$

A.5 Tipos de interfaz de trenes

La definición de reglas completas de formación de subtipos de firma para interfaces de trenes está fuera del ámbito de este modelo de referencia (véase 7.2.4.2). Obsérvese, sin embargo, que un tipo de firma de un flujo individual puede formalizarse interpretándolo en el lenguaje *Type*. Los elementos de *Type* asociados con un flujo pueden definirse por entornos bien formados en el mismo dominio que el conjunto de *Type* definido por la gramática de la Figura A.8, donde la etiqueta a_i corresponde al nombre del flujo.

La regla de formación de subtipos indicada en 7.2.4.2, asociada con flujos correspondientes (suponiendo que tienen la misma causalidad) corresponde justamente a la relación de subtipo, \leq , en este caso.

α	::=	$\langle a_i : Flowsig \rangle$
<i>Flowsig</i>	::=	$Arg \rightarrow Nil$
<i>Arg</i>	::=	$Nil \mid t$

Figura A.8 – Sintaxis abstracta para tipos de firma de interfaz de trenes

A.6 Ejemplo

Considérense las siguientes definiciones de tipo de firma de interfaz de operaciones de servidor (es decir, el entorno bien formado)

$$\Upsilon = \{t \mapsto \alpha, f_t \mapsto \beta\}$$

donde

$$\alpha =_{def} \langle op : t \rightarrow [ok : Nil, nok : Nil], factory : Nil \rightarrow [ok : f_t] \rangle$$

$$\beta =_{def} \langle new : Nil \rightarrow [ok : t] \rangle$$

y donde $t, f_t \in Tvar$; $op, factory, new, ok$ and $nok \in \Lambda$ ($op, factory$ y new son nombres de operación; ok y nok son nombres de terminación).

Intuitivamente, el entorno Υ corresponde a la definición de dos tipos, t y f_t . f_t está equipado con una operación solamente, new , que no toma argumento y retorna una referencia a una instancia de tipo t . Cabe pensar, por ejemplo, que f_t es el tipo de un objeto factoría que crea objetos con una interfaz de tipo t a petición, esto es, para cada invocación de operación nueva. t está equipado con dos operaciones: op y $factory$. La operación op toma como argumento una referencia a una instancia de tipo t : esta es una primera instancia de una definición recursiva. La operación $factory$ no toma argumento y retorna una referencia a una instancia de tipo f_t . Cabe pensar que, por ejemplo, para fines de gestión, cada objeto con una interfaz de tipo t puede, a petición (es decir, al invocarse una operación $factory$), retornar una referencia a la factoría que lo creó. Esta es una segunda instancia de una definición recursiva, ya que la definición de f_t hace referencia a t .

Aplicando la definición de Val anteriormente dada, definiendo $\Upsilon_1 =_{def} \{f_t \mapsto \beta\}$, sobrecargando el signo $=$ y utilizando la regla de equivalencia de tipos E.10, se obtiene lo siguiente:

$$\begin{aligned} Val(t, \Upsilon) &= \mu t. Val(\alpha, \{f_t \mapsto \beta\}) \\ &= \mu t. \langle op : Val(t, \Upsilon_1) \rightarrow [ok : Nil, nok : Nil] \\ &\quad factory : Nil \rightarrow [ok : Val(f_t, \Upsilon_1)] \rangle \\ &= \mu t. \langle op : t \rightarrow [ok : Nil, nok : Nil] \\ &\quad factory : Nil \rightarrow [ok : \mu f_t. Val(\beta, \emptyset)] \rangle \\ &= \mu t. \langle op : t \rightarrow [ok : Nil, nok : Nil] \\ &\quad factory : Nil \rightarrow [ok : \mu \langle f_t \langle new : Nil \rightarrow [ok : t] \rangle \rangle] \rangle \\ &= \mu t. \langle op : t \rightarrow [ok : Nil, nok : Nil] \\ &\quad factory : Nil \rightarrow [ok : \langle new : Nil \rightarrow [ok : t] \rangle] \rangle \end{aligned}$$