Unión Internacional de Telecomunicaciones

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

Z.130

Enmienda 1

(06/2006)

SERIE Z: LENGUAJES Y ASPECTOS GENERALES DE SOPORTE LÓGICO PARA SISTEMAS DE TELECOMUNICACIÓN

Técnicas de descripción formal – Lenguaje ampliado de definición de objetos

---

Lenguaje ampliado de definición de objetos: Técnicas de desarrollo de componentes de soporte lógico distribuido – Bases conceptuales, notaciones y correspondencias tecnológicas

Enmienda 1: Nuevo anexo E – Correspondencia entre eODL y CIDL

Recomendación UIT-T Z.130 (2003) – Enmienda 1

UIT-T

Unión Internacional de Telecomunicaciones

RECOMENDACIONES UIT-T DE LA SERIE Z

LENGUAJES Y ASPECTOS GENERALES DE SOPORTE LÓGICO PARA SISTEMAS DE TELECOMUNICACIÓN

TÉCNICAS DE DESCRIPCIÓN FORMAL	
Lenguaje de especificación y descripción	Z.100–Z.109
Aplicación de técnicas de descripción formal	Z.110–Z.119
Gráficos de secuencias de mensajes	Z.120–Z.129
Lenguaje ampliado de definición de objetos	**Z.130–Z.139**
Notación de prueba y de control de prueba	Z.140–Z.149
Notación de requisitos de usuarios	Z.150–Z.159
LENGUAJES DE PROGRAMACIÓN	
CHILL: el lenguaje de alto nivel del UIT-T	Z.200–Z.209
LENGUAJE HOMBRE-MÁQUINA	
Principios generales	Z.300–Z.309
Sintaxis básica y procedimientos de diálogo	Z.310–Z.319
LHM ampliado para terminales con pantalla de visualización	Z.320–Z.329
Especificación de la interfaz hombre-máquina	Z.330–Z.349
Interfaces hombre-máquina orientadas a datos	Z.350–Z.359
Interfaces hombre-máquina para la gestión de las redes de telecomunicaciones	Z.360–Z.379
CALIDAD	
Calidad de soportes lógicos de telecomunicaciones	Z.400–Z.409
Aspectos de la calidad de las Recomendaciones relativas a los protocolos	Z.450–Z.459
MÉTODOS	
Métodos para validación y pruebas	Z.500–Z.519
SOPORTE INTERMEDIO	
Entorno del procesamiento distribuido	Z.600–Z.609

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

Recomendación UIT-T Z.130

Lenguaje ampliado de definición de objetos: Técnicas de desarrollo de componentes de soporte lógico distribuido – Bases conceptuales, notaciones y correspondencias tecnológicas

Enmienda 1

Nuevo Anexo E – Correspondencia entre el eODL y el CIDL

Resumen

En esta enmienda se suministra un ejemplo de correspondencia entre el eODL de la UIT para las especificaciones de componente independiente de la tecnología y un lenguaje dependiente de la tecnología, a saber el CIDL (lenguaje de definición de implementación de componente del OMG, como parte de CORBA 3.0). En esta enmienda se transforma (mediante diversas correspondencias) el concepto de componente, partiendo del diseño y de la implementación (donde son bien conocidos los módulos) hasta el software binario. La composición de los componentes tiene lugar durante el tiempo de ejecución.

Orígenes

La enmienda 1 a la Recomendación UIT-T Z.130 (2003) fue aprobada el 13 de junio de 2006 por la Comisión de Estudio 17 (2005-2008) del UIT-T por el procedimiento de la Recomendación UIT-T A.8.

PREFACIO

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

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

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

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

NOTA

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

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

PROPIEDAD INTELECTUAL

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

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

ÍNDICE

Recomendación UIT-T Z.130

Lenguaje ampliado de definición de objetos: Técnicas de desarrollo de componentes de soporte lógico distribuido – Bases conceptuales, notaciones y correspondencias tecnológicas

Enmienda 1

Nuevo anexo E – Correspondencia entre eODL y CIDL

1) Reemplácense en el Resumen los siguientes ítems

- El anexo D contiene una referencia de soporte lógico para la representación en XML [12] del metamodelo eODL con arreglo al formato de intercambio meta de XML (XMI) [6]. Se facilita en un fichero independiente para poder importar y procesar el metamodelo eODL con las herramientas UML.
- La cláusula 1 presenta una sinopsis de cómo deben utilizar el eODL los diseñadores, implementadores y gestores de sistemas distribuidos. El apéndice I proporciona un ejemplo concreto de su uso.

por:

- El anexo D contiene una referencia de soporte lógico para la representación en XML [12] del metamodelo eODL con arreglo al formato de intercambio meta de XML (XMI) [6]. Se facilita en un fichero independiente para poder importar y procesar el metamodelo eODL con las herramientas UML.
- El anexo E contiene las reglas de correspondencia entre el eODL independiente de la tecnología y el CIDL específico a ella [7].
- En el apéndice I se proporciona una sinopsis de cómo deben utilizar el eODL los diseñadores, implementadores y gestores de sistemas distribuidos. En el apéndice I se suministra un ejemplo de su uso concreto.

2) Actualícese el índice

Añádase lo siguiente al índice, antes del apéndice I. Actualícese la numeración de las páginas con arreglo al nuevo documento.

3) Añádase antes del apéndice I

Anexo E

Correspondencia del eODL con el CIDL

E.1 Introducción

El desarrollo de software basado en componentes es una estrategia enfocada a obtener un software modular y basado en el modelo. Varias correspondencias soportan este tipo de desarrollo, que transforma **modelos de componente** (vistos desde diversas perspectivas, como por ejemplo la de diseño y la de implementación) en **componentes de software** binarios. La composición de los componentes de software tiene lugar durante el tiempo de ejecución.

El **eODL** es un lenguaje que permite la utilización de conceptos para una **descripción de modelo independiente de la tecnología** de los componentes durante su vida útil, desde diversas perspectivas. Algunos conceptos tales como objeto computacional, componente, interfaz, módulo, señal y tipo de datos son esenciales en las perspectivas computacional y de implementación. Además, hay otros conceptos que sirven para la descripción de los entornos de funcionamiento y la utilización de componentes de software.

El **CCM** (CORBA component model [7]) es una norma del OMG para un marco dependiente de la plataforma. Esta norma proporciona un metamodelo para la descripción de componentes CORBA **dependientes de la tecnología**, y la tecnología y el entorno de funcionamiento para componentes desarrollados utilizando dicho metamodelo. El CCM se basa en tecnologías de CORBA ampliamente utilizadas, como por ejemplo el protocolo GIOP y vinculaciones de lenguaje para lenguaje de implementación. En el modelo de componente de CCM se definen dos tipos de interacción de componente, a saber una interacción de tipo RPC con petición/respuesta y una de tipo señal con eventos. Para cada una de estas interacciones, los componentes pueden declarar su utilización o su configuración. El CCM utiliza el lenguaje CIDL para la implementación de la notación de componente.

Con el fin de **cerrar la brecha** entre modelos de componentes de software independientes de la tecnología, presentados como especificaciones eODL y modelos dependientes de la tecnología, presentados como modelos CIDL, se necesitan correspondencias que permitan la transformación automática de modelos.

El lenguaje de definición de implementación de componente (**CIDL**, *component implementation definition language*) del OMG se utiliza para describir la estructura y el estado de las implementaciones de componente CORBA. Los compiladores que aceptan componentes generan estructuras de implementación a partir de las definiciones CIDL. Los constructores de componentes amplían dichas estructuras, creando así implementaciones completas.

Este anexo define las reglas para la correspondencia entre el eODL y el CIDL. Dichas reglas se verifican mediante una implementación de compilador.

E.2 Correspondencia restringida entre el eODL y el CIDL

La definición del eODL se basa ampliamente en conceptos definidos por el IDL 2.x de CORBA [5]. De igual manera, el metamodelo del eODL forma una extensión del metamodelo de CORBA. Si bien los conceptos adaptados se atribuyen a la *perspectiva computacional* del eODL, desafortunadamente el metamodelo de CCM no soporta los conceptos eODL de la *perspectiva de despliegue* y del *entorno objetivo*. El metamodelo MOF del CCM no define aún dicho campo. Sólo

existen tipos de documentos XML definidos que son necesarios para la puesta en marcha final de la arquitectura de despliegue.

Conclusión: No se hacen corresponder los conceptos eODL que tienen que ver con la *perspectiva de despliegue* y la *perspectiva de entorno objetivo*. Conviene ampliar las reglas de correspondencia, en espera de que el OMG finalice el proceso de normalización.

E.3 Correspondencia de conceptos eODL que son conceptos CORBA en el CCM

Tal como el metamodelo eODL, el metamodelo CCM también amplía los conceptos CORBA 2.x. Por consiguiente, para los conceptos eODL que se derivan de CORBA se escoge la correspondencia más simple, a saber la **correspondencia idéntica**. De esta manera, es posible atribuir conceptos básicos de la *perspectiva computacional* de eODL, como **tipos de datos, interfaces, operación** y **atributos**, del nivel independiente de la plataforma al nivel específico de la plataforma, solicitando así el trabajo de quien desarrolla.

Al utilizar el metamodelo CORBA como fuente y como destino para la correspondencia, es posible que ocurran traslapos al definir las reglas de transformación. Esto sólo ocurre, como consecuencia de la correspondencia idéntica, cuando se emplean conceptos del metamodelo CORBA en un contexto que no se deriva de él mismo.

E.4 Correspondencia de conceptos desde el punto de vista computacional

E.4.1 Señal

Las señales llevan información en el eODL y son transportadas durante una interacción que se basa en señal desde el remitente hasta el receptor.

Regla 1: Para cada **SignalDef** en eODL se crea un **EventDef** en CCM con el mismo nombre. Los nombres y tipos de datos correspondientes en un **CarryField** en el eODL se hacen corresponder con los elementos **ValueMemberDef** en CCM, que están incluidos dentro del **EventDef**. Todo **ValueMemberDef** creado tiene visibilidad pública (isPublicMember==true).

Ejemplo:

```
signal Sig {
  long l;
};
```

Se hace corresponder al CIDL

```
eventtype Sig {
  public long l;
};
```

E.4.2 Consumir y producir

Se supone que los elementos de interacción consumir y producir del eODL definen la interacción basada en señal dentro de una interfaz. Si bien dicha interacción existe en el CCM (***EventDef***), no se permite que sea parte de una interfaz, sino que el CCM la define solamente como una parte directa de una definición de componente. Esto tampoco está permitido en el eODL: sólo se permiten atributos. Si se prohibiese completamente los elementos consumir y producir en el modelo eODL, no sería posible tener una interacción basada en señal. Por lo tanto, se define una construcción que las reemplaza, que aunque incrementa la complejidad de la correspondencia, por lo menos permite dichas interacciones. En el CCM la definición de una señal (***EventDef***) forma una definición de una interfaz para el intercambio de señales. Al definir puertos de componente, se tratan como puertos propios. Es decir, para cada elemento de interacción basado en del eODL se define un puerto aparte en el componente.

Regla 2: No se hacen corresponder los elementos de tipo ***ConsumeDef*** y ***ProduceDef*** en el eODL, pero su procesamiento se efectúa siguiendo las reglas para puertos.

Ejemplo:

```
signal Sig;
interface A {
  consumes Sig c;
  produces Sig p;
};
```

Se hace corresponder al CIDL,

donde se suprimen los elementos de interacción basados en señal del ***EnhancedInterfaceDef****. Sólo se reflejan las señales en CCM como* ***EventDef****.*

```
eventtype Sig {
};
interface A {
};
```

E.4.3 Medios, sumidero y fuente

El CCM soporta las interacciones operacionales y basadas en señal. **No se reflejan** las interacciones basadas en el tren. Si bien se está realizando algún trabajo tendiente a ampliar el CCM a dichos conceptos, el resultado aún no se materializa en la norma.

Por consiguiente, la correspondencia no transformará los elementos de modelo de dichos conceptos del eODL al CCM.

E.4.4 Tipos de CO, soporta y requiere

Tanto el eODL como el CCM son ampliaciones de conceptos de CORBA e introducen el concepto de componente. En el caso del eODL, dicho concepto se denomina el tipo CO, mientras que en el CCM se le llama componente. En otras palabras, se hace corresponder un tipo CO con un componente. Sólo se permiten las interacciones del tipo CO a través de puertos, puesto que dicha variante de interacción también existe en el CCM.

Regla 3: Para cada ***COTypeDef*** en el eODL se crea un ***ComponentDef*** en el CCM con el mismo nombre. Si el ***COTypeDef*** B especializa el ***COTypeDef*** A en el eODL, el correspondiente ***ComponentDef*** B especializa el ***ComponentDef*** A en el CCM. No se hacen corresponder las relaciones ***supports*** y ***requires****.* En el eODL no se permite la herencia múltiple de ***COTypeDef****.*

Ejemplo:

```
CO A {
};
CO B {
};
```

Se hacen corresponder los tipos CO del eODL con componentes en el CCM, a la vez que se conserva la relación de herencia.

```
component A {
};
component B : A {
};
```

E.4.5 Home (*HomeDef*)

Existe un concepto más en el CCM, que se relaciona estrechamente con el componente CCM. Se trata del concepto home, que se utiliza para la gestión de los componentes durante la ejecución. Un home suministra una facilidad para crear ejemplares de los componentes, por lo que una definición de componente sin home es incompleta. La correspondencia crea entonces para cada tipo CO también un *home* en el CCM.

Regla 4: Para cada ***COTypeDef*** en el eODL se crea un ***HomeDef*** en el CCM, y su nombre se construye concatenando el nombre del ***COTypeDef*** y "**`_Home`**". El ***ComponentDef*** y el ***HomeDef*** resultantes participan en la asociación ***Component_Home***. Si un ***COTypeDef*** B especializa un ***COTypeDef*** A en el eODL, el correspondiente ***HomeDef*** B especializa el ***HomeDef*** A en el CCM.

Ejemplo:

```
CO A {
};
CO B : A {
};
```

La correspondencia crea, para cada tipo de CO, un tipo de componente en el CCM, así como un home.

```
component A {
};
home A_Home manages A {};
component B : A {
};
home B_Home : A_Home manages B {};
```

E.4.6 Puerto suministrado y puerto utilizado

Se utilizan los conceptos ***ProvidePortDef*** y ***UsedPortDef*** en el eODL para definir la interfaz entre el tipo CO y el entorno. El concepto correspondiente en el CCM es el puerto (***ComponentFeature***), que aparece en diferentes especializaciones. Existe un puerto para la realización y utilización de una interfaz en el contexto de CORBA, así como uno para la interacción basada en señal.

Regla 5: Para cada ***ProvidePortDef*** en el eODL con un ***COTypeDef***, se crea un ***ProvidesDef*** del ***ComponentDef*** correspondiente en el CCM, con el mismo nombre. La ***InterfaceDef*** referenciada conforme al ***ProvidePortDef*** es parte de la asociación ***provides*** del ***ProvidesDef***. El ***ProvidesDef*** cumple la función ***facet*** con respecto al ***ComponentDef*** dentro del grupo ***Component_Facet***. No se permiten los puertos múltiples (*multiple==true*).

Ejemplo:

```
interface A {
};
CO C {
  provides A the_a;
};
```

Se hacen corresponder los tipos de interfaz proporcionados del tipo CO, sin interacción basada en señal, con puertos simples.

```
interface A {
};
component C {
  provides A the_a;
};
```

Regla 6: Para cada ***UsedPortDef*** en el ODL con un ***COTypeDef***, se crea un ***UsesDef*** del ***ComponentDef*** correspondiente en el CCM, con el mismo nombre. La ***InterfaceDef*** a la que se hace referencia conforme al ***UsedPortDef*** es parte de la asociación ***uses*** del ***UsesDef***. El ***UsesDef*** cumple la función ***receptacle*** con respecto al ***ComponentDef*** dentro del grupo ***Component_Receptacle***. Se hacen corresponder varios puertos en el eODL (*multiple==true*) con varios puertos (*multiple==true*).

Ejemplo:

```
interface A {
};
CO C {
  uses multiple A the_a;
};
```

*Contrario a la correspondencia de **ProvidesDef**, aquí se permite la utilización de puertos múltiples como puertos utilizados.*

```
interface A {
};
component C {
  uses multiple A the_a;
};
```

E.4.7 Puerto producir y consumir

Al haberse suprimido los elementos de interacción de las interfaces debido a la correspondencia (*Regla 2*), deben definirse reglas adicionales para la correspondencia de la interacción basada en señal con los puertos de las interfaces.

Regla 7: Para cada ***ProduceDef*** en el eODL con una ***InterfaceDef*** a la que se ha hecho referencia mediante un ***ProvidePortDef*** de un ***COTypeDef***, se crea un ***PublishesDef*** cuyo nombre se construye a partir de la concatenación del nombre del ***ProvidedPortDef*** con "_" y el nombre del ***ProduceDef***. El ***PublishesDef*** cumple la función ***publishes*** con respecto a la ***ComponentDef*** dentro del conjunto ***Component_Publishes***.

Ejemplo:

```
signal Sig;
interface A {
  produce Sig p;
};
CO C {
  provides A the_a;
};
```

La utilización de interfaces con la interacción de puerto basada en señal siempre produce un puerto separado, debido a que cada interacción basada en señal ha de corresponder con un puerto aparte.

```
eventtype Sig {
};
interface A {
};
component C {
  provides A the_a;
  publishes Sig the_a_p;
};
```

Regla 8: Para cada ***ConsumeDef*** en el eODL con una ***InterfaceDef*** a la que se hace referencia mediante un ***ProvidePortDef*** de un ***COTypeDef****,* se crea un ***ConsumesDef*** en el CCM cuyo nombre se construye concatenando el nombre del ***ProvidedPortDef*** con "_" y el nombre del ***ConsumeDef***. El ***ConsumesDef*** cumple la función de ***consumes*** con respecto al ***ComponentDef*** dentro del grupo ***Component_Consumes***.

Ejemplo:

```
signal Sig;
interface A {
  consume Sig s;
};

CO C {
  provides A the_a;
};
```

A diferencia del ejemplo anterior, se ha modificado el sentido de la interacción basada en señal, por lo que el puerto de envío es ahora un puerto de recepción.

```
eventtype Sig {
};
interface A {
  consume Sig c;
};
component C {
  provides A the_a;
  consumes Sig the_a_c;
};
```

Regla 9: Para cada ***ProduceDef*** en el eODL con un ***InterfaceDef*** al que se hace referencia mediante un ***UsedPortDef*** de un ***COTypeDef****,* se crea un ***ConsumesDef*** en el CCM cuyo nombre se construye concatenando el nombre del ***ProvidedPortDef*** con "_" y el nombre del ***ConsumeDef***. El ***ConsumesDef*** cumple la función ***consumes*** con respecto al ***ComponentDef*** dentro del grupo ***Component_Consumes***.

Ejemplo:

```
signal Sig;
interface A {
  produce Sig p;
};

CO C {
  use A the_a;
};
```

La correspondencia es la misma de la Regla 7, salvo la del puerto utilizado.

```
eventtype Sig {
};
interface A {
};
component C {
   uses A the_a;
  consumes Sig the_a_p;
};
```

Regla 10: Para cada ***ConsumeDef*** en el eODL con un ***InterfaceDef*** al que se hace referencia mediante un ***UsedPortDef*** de un ***COTypeDef****,* se crea un ***PublishesDef*** en el CCM cuyo nombre se construye concatenando el nombre del ***UsedPortDef*** con "_" y el nombre del ***ConsumeDef***. El ***ConsumesDef*** cumple la función de ***publishes*** con respecto al ***ComponentDef*** dentro del grupo ***Component_Publishes***.

Ejemplo:

```
signal Sig;
interface A {
  consume Sig s;
};
CO C {
  use A the_a;
};
```

Comparado con el ejemplo anterior, se ha modificado el sentido de la interacción basada en señal, por lo que el puerto de envío es ahora uno de recepción.

```
eventtype Sig {
};
interface A {
  consume Sig c;
};
component C {
  uses A the_a;
  publishes Sig the_a_c;
};
```

E.4.8 Atributo

Los tipos CO en el eODL pueden contener solamente ***AttributeDef*** como elementos de interacción. De igual manera, el concepto correspondiente en el CCM puede contener ***AttributeDef*** y, por lo tanto, el ***AttributeDef*** de tipos CO se hace corresponder con el ***AttributeDef*** de tipos componente.

Regla 11: Se permite que el ***COTypeDef*** como especialización de ***InterfaceDef****,* con las restricciones definidas por la Rec. UIT-T Z.130, contenga ***AttributeDef***. Para cada ***AttributeDef*** en un ***COTypeDef*** se crea un ***AttributeDef*** en CCM dentro del correspondiente ***ComponentDef*** en CCM. Se conservan el nombre, la asociación y los atributos del ***AttributeDef****,* conforme a E.1.

Ejemplo:

```
CO C {
  readonly attribute long l;
};
```

En realidad, aunque la correspondencia de los atributos es la idéntica, se utiliza fuera del contexto de CORBA.

```
Component C {
  readonly attribute long l;
};
```

E.5 Correspondencia de los conceptos de la perspectiva de implementación

Los conceptos de la perspectiva de implementación eODL describen, mediante artefactos del lenguaje de implementación, la relación entre los elementos de interacción suministrados por los tipos de interfaz de tipo CO y las realizaciones. Al hacerlo, los artefactos modelan elementos del lenguaje de implementación de la tecnología de componente. En el contexto de lenguajes de implementación orientados al objeto esto suele denominarse clases. El eODL es menos restrictivo al considerar la relación de elementos de interacción y artefactos. En particular, no se impone ninguna restricción a la agrupación de elementos de interacción e interfaces. El CCM también define conceptos necesarios para especificar la estructura de la realización de componente. Con este fin, se utiliza el concepto ***ComponentImplDe***. Puesto que cada componente tiene un home definido, existe un concepto ***HomeImplDef*** para su realización. Asimismo, el CCM define el concepto ***SegmentDef*** para una especificación ulterior de la subestructura. Al utilizar el ***SegmentDef*** se define un artefacto que realiza las interfaces proporcionadas del componente. Es decir, el CCM sólo permite relaciones entre los elementos de interacción y los artefactos según su clasificación en interfaces (véase la figura E.1). Además, se prohíbe la atribución de interfaces utilizadas por el componente a ***SegmentDef***. La conexión de elementos de interacción basada en señal a ***SegmentDef*** en caso de envío no se permite, como tampoco ocurre en caso de recepción. En el CCM debe ser posible la atribución de por lo menos las señales de recepción.

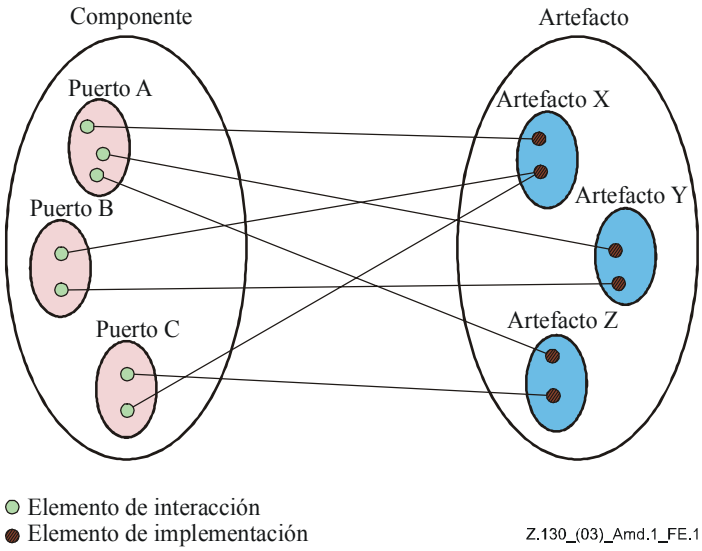

Figura E.1/Z.130 – Atribución de elementos de interacción y artefactos

E.5.1 Artefacto

El eODL utiliza los conceptos ***ArtifactDef*** e ***ImplementationalElementDef*** para describir la estructura de una realización de tipos CO y la atribución de elementos de interacción a los artefactos. El CCM especifica el concepto ***SegmentDef*** para dicha atribución. Por consiguiente, se hacen corresponder los artefactos al ***SegmentDef***. El eODL define para cada artefacto uno de los siguientes patrones de creación de ejemplares

ARTIFACT_PER_REQUEST, ***ARTIFACT_POOL***, ***SINGLETON*** o ***USER_DEFINED***.

Estos patrones describen el tiempo de creación de ejemplar de un artefacto. Los conceptos más cercanos en el CCM son las categorías de tipos de componente ***PROCESS***, ***ENTITY***, ***SESSION*** y ***SERVICE***. De una parte, se refieren solamente a la realización completa de componentes, y no se

da una separación específica para ***SegmentDef***. De otra parte, no se pueden encontrar atribuciones únicas entre patrones de creación de ejemplares y categorías de tipos de componentes (véanse 5.4.1 y 4.1.4 de componentes CORBA).

Regla 12: Para cada ***ArtifactDef*** y el ***ImplementationalElementDef*** contenido en el eODL se crea un ***SegmentDef*** con el mismo nombre en el CCM. Sólo se permite al ***ImplementationElementDef*** contenido dentro del ***ArtifactDef*** relacionarse con elementos de interacción del tipo ***OperationDef*** y ***AttributeDef*** para tipos de interfaz (*Case == supply*) proporcionados a través de puertos. Todos los elementos ***ImplementationElementDef*** dentro de un ***ArtifactDef*** han de cubrir el conjunto completo de elementos de interacción de un ***InterfaceDef***. La asociación ***implementedBy*** contiene entonces en el CCM el ***SegmentDef*** y todos los elementos ***ProvidesDef***, según los diferentes elementos ***InterfaceDef***. Se hace corresponder la atribución ***implemented_by*** del eODL con los ***segments*** de asociación del CCM conforme a los elementos correspondientes creados.

Para todos los elementos ***ArtifactDef*** en el eODL, que están realizando el mismo tipo CO, se crea en el CCM un ***ComponentImplDef***, cuyo nombre se forma concatenando el nombre del ***COTypeDef*** e "**`Impl`**". La categoría de implementaciones de componentes es sesión (*category == SESSION*). El ***ComponentImplDef*** y el ***ComponentDef*** conforme al ***COTypeDef*** participan en la relación ***implemented_by***.

Para cada ***HomeDef*** creado, según el ***COTypeDef*** se crea un ***HomeImplDef***, cuyo nombre se forma concatenando el nombre del ***HomeDef*** e "**`Impl`**". La relación ***implements*** asocia el ***HomeImplDef*** al ***HomeDef*** y la relación ***manages*** asocia el ***ComponentImplDef***.

Ejemplo:

```
interface A {
  void op();
};

CO C {
  provides A the_a;
};
artefact AImpl {
  op implements supply A::op;
};
```

No existe en el eODL un concepto independiente para la implementación de tipo CO aunque el CCM necesita un ComponentImplDef. También se ha de crear un HomeImplDef para el HomeDef creado implícitamente.

```
interface A {
  void op();
};
component C {
  provides A the_a;
};

home C_Home manages C {};
composition session CImpl {
  home executor C_HomeImpl {
    implements C_Home;
    manages CSessionImpl {
      segment AImpl {
        provides facet the_a;
      };
    };
  };
};
```

SERIES DE RECOMENDACIONES DEL UIT-T

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

Impreso en Suiza
Ginebra, 2006