UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

# UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

# Z.100
## Addendum 1
(10/96)

SERIE Z: LENGUAJES DE PROGRAMACIÓN
Lenguaje de especificación y descripción (SDL)

---

## Lenguaje de Especificación y Descripción del CCITT
## Addendum 1

Recomendación UIT-T Z.100 – Addendum 1

(Anteriormente Recomendación del CCITT)

---

RECOMENDACIONES DE LA SERIE Z DEL UIT-T

**LENGUAJES DE PROGRAMACIÓN**

| | |
|---|---|
| **Lenguaje de especificación y descripción (SDL)** | **Z.100–Z.109** |
| Criterios para la utilización y aplicabilidad de técnicas de descripción formal | Z.110–Z.199 |
| Lenguage de alto nivel del UIT-T (CHILL) | Z.200–Z.299 |
| LENGUAJE HOMBRE-MÁQUINA | Z.300–Z.499 |
| 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.399 |
| Varios | Z.400–Z.499 |

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

# PREFACIO

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

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

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución N.° 1 de la CMNT (Helsinki, 1 al 12 de marzo de 1993).

La Recomendación UIT-T Z.100, Addendum 1, ha sido preparada por la Comisión de Estudio 10 (1993-1996) del UIT-T y fue aprobada por la CMNT (Ginebra, 9 al 18 de octubre de 1996).

_____

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.

# ÍNDICE

## SUMARIO

Esta Recomendación es un addendum a la Recomendación Z.100 (1993), y junto con ella describe el lenguaje de especificación y descripción (SDL) del CCITT, constituyendo su versión de 1996. Este Addendum contiene correcciones a la definición de lenguaje anterior y algunas adiciones y cambios al lenguaje. No habrá definición formal adicional de estos cambios.

## ANTECEDENTES

Durante el periodo de estudios de 1993 a 1996, se han identificado algunos errores menores que se han añadido a una "lista básica de cambios" que permanece abierta. Durante el periodo ha habido muchas peticiones para que se mantenga la estabilidad del lenguaje SDL, lo que se ha respetado. Sin embargo, los usuarios han solicitado algunos cambios y adiciones que se han tenido en cuenta. La publicación de un addendum solamente, en lugar de una Recomendación nueva, resalta que la estabilidad se halla implícita en esta versión de lenguaje revisada.

# LENGUAJE DE ESPECIFICACIÓN Y DESCRIPCIÓN DEL CCITT

# ADDENDUM 1

*(Ginebra, 1996)*

## 1 Ámbito

Este addendum define el lenguaje de especificación y descripción SDL del CCITT.

## 2 Referencias

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes.

[1] Recomendación UIT-T Z.100 (1993), *Lenguaje de especificación y descripción del CCITT.*

## 3 Abreviaturas

Este addendum utiliza las siguientes siglas.

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

## 4 Observaciones generales sobre la lista de correcciones a la Recomendación Z.100

Este addendum refleja la lista básica de cambios que se han acumulado durante el periodo de estudios de 1993 a 1996, e incluye todos aquellos que han sido aprobados por la Comisión de Estudio 10 del UIT-T en sus reuniones. La lista básica de cambios incluye entradas según las siguientes categorías:

- *Correcciones* y *clarificaciones*: Consideradas para entrar en vigor inmediatamente y descritas de manera completa.
- *Modificaciones*, *características comprometidas* y *ampliaciones*: Acordadas por la Comisión de Estudio 10 como candidatos para ser incluidas en una revisión de la Recomendación Z.100, pero sujetas a los procedimientos completos de aprobación de la UIT y que solamente entrarán en vigor cuando Recomendaciones nuevas o revisadas sean aprobadas por la UIT.
- *Ítems abiertos* e *ítems cerrados*: Recogen el estado actual de los temas en estudio.

Hay algunas *modificaciones*, *ampliaciones* e *ítems abiertos* que quedan en estudio.

La terminología y directrices de mantenimiento, así como los procedimientos de petición de cambios en el SDL pueden encontrarse en la cláusula 15 de este addendum.

## 5 Correcciones y clarificaciones a la Recomendación Z.100

**Subcláusula 2.2.2**

Cambiar la regla de producción para <path item> en la página 17 para la gramática textual concreta añadiendo la opción de "<operator name> <exclamation>":

```
<path item> ::=
                    <scope unit kind>
                    { <name> | <quoted operator> | <operator name> <exclamation> }
```

**Subcláusula 2.2.2**

Añadir en *Semántica,* al final de la subcláusula antes de los dos últimos párrafos:

Lo que viene a continuación clarifica las reglas de visibilidad para procedimientos, servicios y tipos de servicios (al final de la página 19):

1) Los procedimientos, servicios y tipos de servicios definidos localmente para un proceso pueden acceder y usar libremente las variables y señales de ese proceso.

2) Los procedimientos, servicios y tipos de servicio definidos globalmente no deben utilizar ninguna variable definida en el contexto llamante.

3) Para acceder y utilizar entidades definidas localmente para el proceso llamante, deben utilizarse los parámetros de contexto correspondientes.

**Subcláusula 2.4.5**

Modificar la última frase del penúltimo párrafo de la *Gramática textual concreta* (página 38, primer y segunda líneas) para que diga:

Un procedimiento exportado definido en el proceso que lo contiene debe mencionarse por una entrada o conservación en un servicio exactamente.

**Subcláusula 2.4.5**

En el área de texto de servicio, se ha omitido <valid input signal set>. Se ha incluido en la representación textual y en la representación gráfica para diagrama de proceso correspondiente (<process text area>). La regla de producción corregida es:

```
<service text area> ::=
        <text symbol> contains {
          [ <valid input signal set> ]
          { <variable definition>
                    ....
          | <macro definition>  } *
                              }
```

**Subcláusula 2.4.6**

En el cuerpo de procedimiento, la representación textual tiene un error, mientras que la representación gráfica tiene la producción intuitivamente correcta. La producción correcta debe ser:

```
<procedure body> ::=
        [ <start> ]   { <state> | <free action> } *
```

**Subcláusula 2.7.5**

El primer párrafo después de las producciones para la *gramática textual concreta*, en la página 68 para *Decision,* debe cambiarse de manera que diga (cambios subrayados):

Una <answer part> o <else part> en una decisión o en una opción es una <answer part> o <else part> de terminación, respectivamente, si contiene una <transition> en la cual se especifica una <terminator statement>, o contiene una <transition string> cuya última <action statement> contiene una decisión o una opción de terminación. Una <decision> u <option transition> es una decisión de terminación y opción de terminación, respectivamente, si cada <answer part> y <else part> en el <decision body> de terminación es una <answer part> o <else part> respectivamente.

El primer párrafo del *Modelo,* en la página 69 para *Decision,* debe cambiarse a:

Si una <decision> no es de terminación, es entonces sintaxis derivada para una <decision> en la cual todas las <answer part> y la <else part> de no terminación han insertado al final de su <transition> una <join> con la primera <action statement> que sigue a la decisión o, si la decisión es la última <action statement> en una <transition string>, con el siguiente <terminator statement>.

**Subcláusula 4.10**

El modelo de entrada prioritaria se modifica. La sexta frase del segundo párrafo de la página 105 se modifica como sigue (cambios subrayados):

> Las entradas prioritarias al estado original están conectadas al primer estado, mientras que todas las entradas, incluidas las entradas prioritarias está conectadas al segundo estado (State 2). Todas las demás entradas distintas de las entradas prioritarias son conservadas en el primer estado.

**Subcláusula 4.14**

El no terminal <end> es eliminado en la producción corregida (página 113):

```
<remote procedure save area> ::=
        <save symbol> ***contains***
        {[ <virtuality> ] procedure <remote procedure identifier list> }
```

**Subcláusula 5.3.1.11**

(página 139) en el *Ejemplo* de *Modelo*, la parte axiomas debe ser (suprimir un ")"):

```
        axioms
                Exor (a,b) == (a and (not b)) or ((not a) and b);
```

**Subcláusula 5.3.2**

(página 147) el no terminal <sort> sustituye a extended sort del modo siguiente:

```
<operator result> ::=
                returns [ <variable name> ] <sort>
```

**Subcláusula 5.4.6**

La regla para <external properties> debe ser:

```
<external properties> ::=
                alternative
                        <external formalism name> [ , <word> ] <end>
                        <external data description>
                [ endalternative ] <end>
```

**Subcláusula 6.1.2**

(expresión de tipo) en la lista de casos de *Semántica*, las palabras "o definición de visión" del cuarto ítem deben suprimirse ya que no se aplican.

**Subcláusula 6.2**

Cambiar el penúltimo párrafo de la *gramática textual concreta* (página 178) por:

> La vinculación de una variable real o de un parámetro de contexto de sinónimo con su definición no está resuelta por el contexto sino utilizando las reglas de visibilidad.

**Subcláusula 6.3.2**

La Recomendación Z.100 define cuándo los tipos virtuales son permitidos dentro de un bloque virtual, y la limitación implícita para cada caso. Algunas situaciones no claras han sido señaladas. Para clarificar este tema, se proponen las dos frases siguientes como adiciones a 6.3.2.

Adición a las condiciones estáticas de la *gramática textual concreta:*

> Si se utilizan **inherits** y **atleast** entonces el tipo heredado debe ser idéntico a la limitación o ser un subtipo de ésta.
>
> En el caso de una limitación implícita, la redefinición que envuelve **inherits** debe ser un subtipo de la limitación.

# 6 Ampliaciones menores a la Recomendación Z.100

**Subcláusula 2.2.2**

Añadir al final del párrafo que comienza con "Se permite omitir ...":

> Con respecto a la visibilidad, si el <identifier> no contiene un <qualifier>, una <package reference clause> se considera como la unidad de ámbito envolvente más próxima a la unidad de ámbito a la que está asociada y contiene las entidades visibles desde el paquete.

**Subcláusula 2.2.6**

Se añadirá un símbolo de comentario adicional como una opción que requiere un cambio en la *Gramática gráfica concreta:*

<comment area> ::=

{<comment symbol> | <comment symbol2>} ***contains*** <text>

***is connected to*** <dashed association symbol>

Además, añadir después la definición del <comment symbol>:

<comment symbol2> ::=

# 7 Armonización de señales y entidades remotas

## 7.1 Motivo

Las definiciones de procedimiento remoto y las definiciones de variable remota sirven para el mismo propósito que las definiciones de señal: definir las primitivas de comunicación. Sin embargo, al comparar con las señales, hay en SDL-92 dos limitaciones en el uso de procedimientos remotos y variables remotas:

1) No pueden ser mencionados en las interfaces (rutas de señal, canales y puertas).

2) No pueden estar comunicados con el entorno del sistema.

En el SDL-96 los procedimientos remotos y las variables remotas están armonizados con las señales. Esto proporciona una mayor potencia expresiva al SDL y da un esquema más coherente para la especificación de las interfaces (y hace por tanto al SDL menos complejo).

La posibilidad de comunicar con el entorno mediante el uso de llamadas a procedimiento remoto es necesaria por dos razones:

1) Cuando se construyen sistemas grandes debe ser posible contemplar una parte del sistema como un sistema propio a fin de aplicar adecuadamente los principios de prueba, reutilización y encapsulación. Tal parte será típicamente una subestructura de bloque, en cuanto que aparece como un sistema completo, pero será una problema importante contemplar una subestructura de bloque como un subsistema, porque las subestructuras de bloques, al contrario de los sistemas, pueden comunicar con su entorno utilizando procedimientos y variables remotos.

2) Cuando se utiliza SDL para arquitecturas de computación distribuida, tales como la definida por el consorcio TINA, el número de servidores (que pueden ser sistemas SDL) que ofrecen interfaces a clientes (que pueden también ser sistemas SDL) puede cambiar durante la ejecución. La utilización de SDL-92 en este área es difícil.

## 7.2 Solución

Los procedimientos remotos y las variables remotas pueden especificarse en listas de señales, de modo que ellas, en cuanto señales, pueden ser especificadas en canales, rutas de señales, puertas, conjunto de señales y conservación. Las unidades de ámbito que tienen procedimientos remotos y variables remotas en las puertas, canales o rutas de señales conectadas, necesitan incluir <imported procedure specification> e <imported variable specification>. Para comunicar con el entorno, las entidades remotas deben mencionarse en los canales conectados al entorno del sistema.

Asimismo, <basic input part> es armonizada con <basic save part> del mismo modo que una <basic input part> en SDL-96 consiste en una lista de <signal list item>. Dicho de otro modo, puede mencionarse un identificador de lista de señales en una entrada de SDL-96. Las dos extensiones juntas eliminan la necesidad de las producciones <remote procedure save> y <remote procedure input>.

Si un identificador en un <signal list item> puede denotar tanto una señal como un procedimiento/variable remoto de acuerdo con las reglas de visibilidad, el identificador denota la señal, a menos que el identificador vaya precedido por la palabra clave **procedimiento** (para un procedimiento remoto) o **remoto** (para una variable remota).

## 7.3 Ejemplo

**signal** s1, s2(integer);

**remote procedure** rp;

**signallist** slist = s1, s2, **procedure** rp;  /* La palabra clave **procedure** no es necesaria si la señal no visible es denominada rp */

...

**gate** g **in with** (slist);

...

**input** (slist);

...

## 7.4 Cambios

**Subcláusula 1.3.2**

En la primera línea: cambiar "recibidos del" por "intercambiados con el".

En la tercera línea: cambiar "las señales" por "los estímulos".

**Subcláusula 2.1**

En la línea 2: cambiar "por señales" para que diga "por medio de señales y procedimientos/variables remotos".

**Subcláusula 2.4.1.2**

En la producción <entity kind>, cambiar

| procedure

en

| [remote] procedure

**Subcláusula 2.4.1.2**

Añadir, después de la producción para <interface>:

> **procedure** se utiliza para la selección tanto de procedimientos (normales) como de procedimientos remotos. Si tanto un procedimiento normal como un procedimiento remoto tienen el <name> dado, **procedure** denota el procedimiento normal. Para obligar a que <definition selection> denote el procedimiento remoto, la palabra clave **procedure** puede ser precedida por **remote.**

**Subcláusula 2.4.2**

En *Semántica* tercera línea: sustituir "sólo puede efectuarse mediante señales. Dentro de un sistema, estas señales son transportadas por canales" por "sólo puede efectuarse mediante señales, procedimientos remotos y variables remotas. Dentro de un sistema, estas entidades son transportadas por canales explícitos o implícitos".

**Subcláusula 2.4.4**

Página 36, línea 5 a partir del principio: sustituir "ninguna señal" por "ningún estímulo"

Página 36, añadir después del párrafo que comienza con "Las señales recibidas por instancias de proceso ...":

> "Las llamadas a procedimiento remoto llamante y sirviente y las variables remotas que acceden, también corresponden a intercambio de señales, véanse 4.13 y 4.14".

**Subcláusula 2.5.1**

Primera línea de *Semántica:* sustituir "señales" por "señales (incluidas las señales implícitas supuestas a partir de procedimientos remotos y variables remotas, véanse 4.13 y 4.14)."

**Subcláusula 2.5.2**

Primera línea de *Semántica:* sustituir "señales" por "señales (incluidas las señales implícitas supuestas a partir de procedimientos remotos y variables remotas, véanse 4.13 y 4.14)".

**Subcláusula 2.5.2**

Añadir un párrafo en *Semántica:*

> Los procedimientos remotos y las variables remotas no necesitan mencionarse en las listas de señales de los canales y rutas de señales entre el importador y el exportador, pero si una variable remota es mencionada en un canal o ruta de señales, entonces debe mencionarse en el trayecto de comunicación completo entre el importador y el exportador. Un procedimiento o variable importado es mencionado como saliente del importador y un procedimiento o variable exportado es mencionado como entrante al exportador. Si un procedimiento o variable importado no es mencionado en la ruta de señales saliente de un proceso o servicio o mencionado en las puertas de un tipo de proceso o tipo de servicio, entonces debe ser especificado en una <imported procedure specification> o <imported variable specification> en el (tipo de) proceso o (tipo de) servicio.

Página 47, tercera línea del segundo párrafo, cambiar:

> "<output> y" por "<output>, <exported variable definition>, <procedure preamble> <imported procedure specification>, <imported variable specification> y"

Página 47, tercera línea del tercer párrafo, cambiar:

> "<output> y" por "<output>, <exported variable definition>, <procedure preamble>, <imported procedure specification>, <imported variable specification> y"

Página 47, segunda línea del quinto párrafo, cambiar:

> "output> y " por "<output>, <exported variable definition> <procedure preamble>, <imported procedure specification>, <imported variable specification> y"

Página 47, tercera línea del sexto párrafo, cambiar:

> "<output> y" por "<output>, <exported variable definition>, <procedure preamble>, <imported procedure specification>, <imported variable specification> y"

Página 47, tercera línea del séptimo párrafo, cambiar:

> "una salida" por "las salidas, especificaciones de procedimiento importadas y especificaciones de variable importadas"

Página 47, tercera línea del párrafo 8, cambiar:

"y las <output> y" por "las <output>, <exported variable definition>, <procedure preamble> <imported procedure specification>, e <imported variable specification>"

**Subcláusula 2.5.5**

Cambiar "identificadores de señal" por "identificadores de señal, procedimientos remotos, variables remotas"

**Subcláusula 2.5.5**

Cambiar <signal list item> y el párrafo siguiente en *Gramática textual concreta:*

<signal list item> ::=

<signal identifier>

| (<signal list identifier>)

|<timer identifier>

| [**procedure <**remote procedure identifier>

| [**remote** <remote variable identifier>

La <signal list> que se construye reemplazando todos los <signal list identifier> de la lista por la lista de los <signal identifier> o <timer identifier> que éstos denotan y reemplazando todos los <remote procedure identifier> y todos los <remote variable identifier> por una de las señales implícitas que cada uno denota (véanse 4.13 y 4.14), corresponde a un *Signal-identifier*-**set** en la *Gramática abstracta.*

Un <signal list item> que es un <identifier> denota un <signal identifier> o un <timer identifier> si ello es posible de acuerdo con las reglas de visibilidad, en otros casos un <remote procedure identifier> si ello es posible de conformidad con las reglas de visibilidad y en otros casos un <remote variable identifier>. Para obligar a un <signal list item> para que denote un <remote procedure identifier> o un <remote variable identifier> pueden utilizarse las palabras clave **procedure** o **remote**.

**Subcláusula 2.6.4**

En *Gramática textual concreta* suprimir <input part> y renombrar <basic input part> como <input part>.

Cambiar *Gramática textual concreta* para <stimulus> a

<stimulus> ::=

<signal list item> [**(**[<variable>] {**,**[<variable>]}***)**]

Una <signal list item> no debe denotar un <remote variable identifier> y si denota un <remote procedure identifier> o un <signal list identifier>, los parámetros de <stimulus> (incluidos los paréntesis) deben ser omitidos.

**Subcláusula 2.6.4**

Añadir a *Modelo:*

Un <stimulus> cuyo <signal list item> es un <signal list identifier>, es sintaxis derivada para una lista de <stimulus> sin parámetros e insertada en la <input list> o <priority input list> envolvente. En esta lista hay una correspondencia entre los <stimulus> y los miembros de la lista de señales.

**Subcláusula 2.6.4**

Añadir después del primer párrafo en *Gramática textual concreta:*

En la *Gramática abstracta*, los <remote procedure identifier> son también representados como *Signal-identifiers*.

**Subcláusula 2.6.4**

Suprimir <input part> y renombrar <basic input part> como <input part>.

**Subcláusula 2.6.4**

Suprimir <input area> y renombrar <basic input area> como <input area>.

**Subcláusula 2.6.5**

Primer párrafo: cambiar "identificadores de señal" por "identificadores de señal e identificadores de procedimiento remoto".

**Subcláusula 2.6.5**

Suprimir <save part> y renombrar <basic save part> como <save part>.

**Subcláusula 2.6.5**

Suprimir <save area> y renombrar <basic save area> como <save area>.

**Subcláusula 4.2.2**

Suprimir <remote procedure input area> y <remote procedure save area> como alternativas en<any area>

**Subcláusula 4.13**

Cambiar <import expression> por

<import expression> ::=

**import (** <remote variable identifier> [**,**<destination>] [ **via** <via path>]**)**

**Subcláusula 4.13**

*Modelo*, primera línea: cambiar "se transportan por canales y rutas de señales implícitos" por "se transportan por canales y rutas de señales implícitos y explícitos. Los canales y rutas de señales son explícitos si la variable remota ha sido mencionada en la <signal list> (saliente para el importador y entrante para el exportador) de al menos una puerta, canal o ruta de señales conectada al importador o exportador. Cuando una variable remota es transportada por canales y rutas de señales explícitos, la palabra clave **nodelay** procedente de la <remote variable definition> es ignorada.

**Subcláusula 4.14**

*Gramática textual concreta:* suprimir <remote procedure input transition> y <remote procedure save>

**Subcláusula 4.14**

Cambiar <remote procedure call body> por:

<remote procedure call body> ::=

<remote procedure identifier> [<actual parameters>]

[**to** <destination>] [**via** <via path>]

**Subcláusula 4.14**

*Gramática textual concreta*, añadir:

Cuando la <destination> y el <via path> son omitidos, hay ambigüedad sintáctica entre <remote procedure call body> y <procedure call>. En este caso, el <identifier> contenido denota un <procedure identifier> si ello es posible de acuerdo con las reglas de visibilidad y en los demás casos un <remote procedure identifier>.

**Subcláusula 4.14**

*Gramática gráfica concreta:* suprimir <remote procedure input area> y <remote procedure save area>

**Subcláusula 4.14**

Último párrafo antes de *Modelo:* cambiar dos ocurrencias de <remote procedure save> en <save> y dos ocurrencias de <remote procedure input transition> en <input part>.

**Subcláusula 4.14**

*Modelo:* cambiar "se transportan por canales y rutas de señales implícitos." por "se transportan por canales y rutas de señales implícitos y explícitos." Los canales y rutas de señales son explícitos si el procedimiento remoto ha sido mencionado en la <signal list> (la saliente para el importador y la entrante para el exportador) de al menos una puerta, canal o ruta de señales conectada al importador o exportador. Cuando un procedimiento remoto es transportado por canales y rutas de señales explícitos, la palabra clave **nodelay** procedente de la <remote procedure definition> es ignorada.

**Subcláusula 4.14**

Página 114, ítem a): cambiar dos veces "**to** destination" por "**to** destination **via** viapath"

**Subcláusula 4.14**

Página 114: suprimir el párrafo que comienza por "Una <remote procedure input transition> o <remote procedure input area> ...

Suprimir **remote procedure input** y **remote procedure save** del glosario.

# 8 Procedimientos externos y operadores externos

## 8.1 Motivo

El SDL se utiliza tanto como lenguaje de especificación como de implementación. Sin embargo, hay situaciones en las cuales el SDL no debe o no puede utilizarse. Tales situaciones incluyen:

- cuando es conveniente reutilizar algún código existente, escrito en otro lenguaje;
- cuando se establece una interfaz con un programa externo (por ejemplo el API de Windows) y el programa externo no define una vinculación del lenguaje con el SDL (típicamente, un programa externo exige la utilización de C++ o Pascal, si es soportada una interfaz externa totalmente);
- cuando es necesario un código de bajo nivel, como ocurre en el desarrollo de excitadores de dispositivo; o
- cuando se trabaja con punteros.

Permitiendo llamadas en SDL a procedimientos y operadores desarrollados utilizando un lenguaje distinto del lenguaje SDL es posible utilizar el SDL como lenguaje de implementación de manera más general.

## 8.2 Solución

Se propone introducir procedimientos y operadores externos en SDL. Al igual que para sinónimos externos y datos externos, la semántica de llamada en un procedimiento externo no estará en el ámbito del SDL.

Cuando se utiliza SDL como lenguaje de especificación, pueden emplearse procedimientos y operadores externos para expresar que el efecto de una llamada dada no está dentro del ámbito de la especificación. Cuando se utiliza el SDL como lenguaje de implementación, pueden utilizarse procedimientos y operadores externos para establecer la interfaz con otros lenguajes.

El formato de un procedimiento/operador externo debe ser el mismo que el formato de referencia procedimiento/operador, con la palabra clave **referenced** substituida por **external.** No se debe utilizar sintaxis gráfica especial al definir procedimientos externos (como similares a procedimientos remotos), pero, a fin de permitir el chequeo de tipos completa, los parámetros formales y el tipo de resultado deben estar en la definición (éste es ya el caso para referencias de operador en SDL-92).

## 8.3 Ejemplo

SDL no tiene características incorporadas para la ordenación de cadenas de caracteres, de modo que puede añadirse una función biblioteca C en alguna parte:

**procedure** strcmp **fpar** op1, op2 Characterstring; **returns** Integer; **external**;

## 8.4 Cambios

Cambios en la **subcláusula 2.4.6**:

Añadir <external procedure definition> como alternativa en <procedure definition>, es decir,

<procedure definition> ::=

<external procedure definition> |

<procedure preamble>

**procedure** {<procedure name> | <procedure identifier> }

...

**endprocedure** [<procedure name> | <procedure identifier>] <end>

Añadir a *Gramática Textual Concreta:*

<external procedure definition> ::=

**procedure** <procedure name>

[<procedure signature>] **external** <end>

Un procedimiento externo no puede mencionarse en una < type expression>, en un <formal context parameter> o en una < atleast constraint>.

No hay sintaxis gráfica correspondiente para <external procedure definition>.

Añadir a *Semántica:*

Un procedimiento externo es un procedimiento cuyo <procedure body> no está incluido en la descripción de SDL. Conceptualmente, un procedimiento externo se supone que está dado por un <procedure body> y que es transformado en una <procedure definition> como parte de un sistema genérico, véase 4.3.1.

**Subcláusula 5.3.2**

Añadir <external operator definition> como alternativa en <operator definition>, es decir,

<operator definition> ::=

<external operator definition> |

**operator** {<operator identifier> | <operator name> }

...

**endoperator** [<operator identifier> | <operator name>] <end>

Añadir a *Gramática textual concreta:*

<external operator definition> ::=

**operator** <operator name>

[<operator signature>] **external** <end>

No hay sintaxis gráfica correspondiente para <external operator definition>.

Añadir a *Semántica:*

Un operador externo es un operador cuyo comportamiento no está incluido en la descripción de SDL. Conceptualmente, un operador externo se supone que tiene un comportamiento dado y que se transforma en una <operator definition> como parte de la transformación de sistema genérico, véase 4.3.1.

# 9 Ampliaciones y armonización de canales y rutas de señales

Los aspectos relativos a las ampliaciones y armonización de canales y rutas de señales son:

1) Posibilidad de canales opcionales.
2) Simplificación de la especificación de canales y rutas de señales.
3) Canales/rutas de señales conectados a la misma unidad de ámbito en los dos puntos extremos.

## 9.1 Posibilidad de canales opcionales

Del mismo modo que para las rutas de señales, en SDL-96 es opcional especificar los canales para una unidad de ámbito dada.

Los motivos de esta ampliación son:

- En sistemas compuestos por muchos bloques, los canales reducen la legibilidad, lo que hace difícil dividir un sistema/subestructura en varias páginas.
- Para sistemas cliente/servidor abiertos, la atención en las interfaces ofrecido/consumido por el servidor/cliente es mayor que en los enlaces de comunicaciones (dinámicamente cambiantes).

## 9.2 Simplificación de la especificación de canales y rutas de señales

Cuando los canales/rutas de señales se especifican para una unidad de ámbito dada, también puede omitirse alguna de su información "redundante", simplificando con ello los diagramas:

- *El nombre del canal/ruta de señales*

  A menudo el nombre del canal o ruta de señales no aparece en ninguna parte. En particular, es el caso en que se conecta a instancias basadas en tipos porque en tales casos no hay conexiones canal-ruta de señales y el identificador de puerta se utiliza **output via** en lugar del canal/ruta de señales.

- *Las lista de señales*

  A menudo las listas de señales de un canal/ruta de señales pueden derivarse de sus puntos extremos. En particular es el caso en que se conectan instancias basadas en tipos y cuando se hacen conexiones de instancias 1 a 1.

## 9.3 Canales/rutas de señales conectados a la misma unidad de ámbito en los dos puntos extremos

Permitir que un canal o una ruta de señales se conecte a la misma unidad de ámbito en los dos puntos extremos es ya un requisito reconocido de SDL-96 al aparecer listado como característica abierta para SDL-92.

En SDL-96, los canales y rutas de señales unidireccionales (no bidireccionales) pueden conectarse a la misma unidad de ámbito en los dos puntos extremos (que no deben ser **env**).

Tales canales y rutas de señales deben ser unidireccionales para superar los problemas de ambigüedad relativos a los conjuntos de bloques, subestructuras de canal y conexiones a instancias sin tipo.

## 9.4 Cambios

Estos cambios permiten lo siguiente:

- Conectar los dos puntos extremos de un canal o ruta de señales al mismo bloque, conjunto de proceso o servicio.
- Omitir canales desde una unidad de ámbito.
- Omitir el nombre de un canal o ruta de señales a partir de la definición del canal o ruta de señales.
- Omitir una o ambas listas de señales a partir de un canal o ruta de señales.
- Mencionar el mismo nombre de canal o ruta de señales en varios lugares en la frontera de un diagrama de bloque, proceso o servicio.

**Subcláusula 2.4.4**

*Gramática textual concreta:* cambiar el penúltimo párrafo por:

"La utilización de <valid input signal set> se define en 2.5.2, *Gramática textual concreta*, y en 7.3."

**Subcláusula 2.4.5**

*Gramática textual concreta:* sustituir el último párrafo por:

"Los <valid input signal set> (véase 7.3.1) de los servicios dentro de un proceso deben estar separados. Además, dos servicios dentro de un proceso no pueden consumir instancias del mismo tipo de temporizador."

**Subcláusula 2.5.1**

*Gramática abstracta:* en el primer párrafo a continuación de las reglas de gramática, sustituir la última frase por:

"Si los dos puntos extremos son el mismo bloque, el canal debe ser unidireccional (es decir, el segundo *Channel-path* en *Channel-definition* debe estar ausente)."

**Subcláusula 2.5.1**

*Gramática textual concreta:* en la regla de gramática para <channel definition>, sustituir "<channel name>" por "[<channel name>]"

La regla de gramática para <channel path> se modifica así:

<channel path> ::=

**from** <channel endpoint> **to** <channel endpoint>

[ **with** <signal list> ] <end>

Añadir el siguiente párrafo inmediatamente después de las reglas de gramática:

"El < channel name > de terminación solamente puede especificarse si se especifica el <channel name> de arranque. Si no se especifica el <channel name> de arranque, el canal no puede ser referido por el nombre."

Añadir el siguiente párrafo inmediatamente antes de *Gramática gráfica concreta:*

"Un canal puede conectar cada una de las dos direcciones de una puerta bidireccional a la otra."

**Subcláusula 2.5.1**

*Gramática gráfica concreta:* en la regla de gramática para <channel definition area>, "<channel name>" es sustituido por "[<channel name>]" , y la primera ocurrencia de "<signal list area>" se sustituye por"[<signal list area>]".

En el primer párrafo después de las reglas de gramática, sustituir la primera ocurrencia de "un <signal list area>" por "al menos un <signal list area>"

**Subcláusula 2.5.1**

*Semántica:* en el primer párrafo, sustituir la primera frase por:

"Una *Channel-definition* representa una ruta de transporte de señales (incluidas las señales implícitas supuestas a partir del intercambio de procedimientos remotos y variables remotas, véanse 4.13 y 4.14)."

Suprimir el último párrafo.

**Subcláusula 2.5.1**

*Modelo:* sustituir el texto completo por:

"*Nombres de canal implícitos:* Si el <channel name> es omitido de una <channel definition> o <channel definition area>, el canal está nombrado implícita y unívocamente, a menos que el canal tenga una subestructura asociada. En este caso, el <channel name> es el mismo que el <channel substructure name>.

*Canales implícitos y listas de señales implícitas de canales:* Si un sistema, tipo de sistema, subestructura de bloque, o subestructura de canal no contiene canales explícitos, se derivan canales implícitos sin listas de señales. Después, los canales sin listas de canales explícitas son rellenados con señales, procedimientos remotos y variables remotas. Los detalles se describen en 7.3.3 y 7.3.5.

Si un sistema, tipo de sistema, subestructura de bloques o subestructura de canales contiene canales explícitos para señales, pero no contiene canales explícitos para procedimientos remotos y variables remotas, se derivan canales implícitos que pueden cursar los procedimientos remotos y las variables remotas en cuestión. Los detalles se describen en 7.3.4.

Los <channel path> definidos explícitamente deben tener listas de señales no vacías después de estas transformaciones.

*Canales conectados a conjuntos de bloque basado en tipo:* Un canal en el cual los dos puntos extremos son puertas de una <textual typebased block definition> representa canales individuales desde cada uno de los bloques en este conjunto a todos los bloques en el conjunto, incluido el bloque mismo. Cualquier canal bidireccional resultante que conecte un bloque de este conjunto al bloque mismo se divide en dos canales unidireccionales.

Un canal en el cual un punto extremo sea una puerta de una <textual typebased block definition> representa canales individuales hacia o desde cada uno de los bloques del conjunto.

Los canales individuales para cada canal original tienen todos la misma característica de retardo que el canal original."

**Subcláusula 2.5.2**

*Gramática abstracta:* sustituir el segundo párrafo después de las reglas de gramática por:

"Si los dos puntos extremos son el mismo conjunto de procesos o servicio, la ruta de señales debe ser unidireccional (es decir, el segundo *Signal-route-path* en *Signal-route-definition* debe estar ausente)".

Añadir el siguiente párrafo inmediatamente antes de *Gramática gráfica concreta:*

"Una ruta de señales puede conectar cada una de las dos direcciones de una puerta bidireccional a la otra."

**Subcláusula 2.5.2**

*Gramática textual concreta:* en la regla de gramática para <signal route definition>, "<signal route name>" es sustituido por "[<signal route name >]".

La regla de gramática para <signal route path> se cambia por:

<signal route path> ::=

**from** <signal route endpoint> **to** <signal route endpoint>

[ **with** <signal list> ] <end>

Añadir el siguiente párrafo inmediatamente después de las reglas de gramática:

"Si no se especifica el <signal route name>, no es posible referirse a la ruta de señales por el nombre."

Añadir el siguiente texto inmediatamente antes de *Semántica:*

"Las siguientes reglas se aplican si la <block definition> o la <block type definition> directamente que incluye una <process definition> no contiene <signal route definition>, o si algunos de los <signal route path> que conducen a la <process definition> no contienen <signal list>:

- Si la <process definition> no tiene servicios, debe contener un <valid input signal set> explícito.
- Si la <process definition> contiene servicios, cada <service definition> debe o bien contener un <valid input signal set> explícito, o todos los <signal route path> que conducen a la <service definition> deben contener <signal list> explícitas.

Una <service definition> debe contener un <valid input signal set> si la <process type definition> o <process definition> circundante no contiene <signal route definition>, o si alguno de los <signal route path> que conducen a la <service definition> no contiene <signal list>.

Si el <valid input signal set> de una <process type definition>, <process definition>, <service type definition> o <service definition> se especifica explícitamente, se aplican las siguientes reglas:

- El <valid input signal set> debe ser un subconjunto del <valid input signal set> completo (véase 7.3.1).

- En el caso de un tipo de proceso o tipo de servicio, el <valid input signal set> no necesita mencionar señales que están explícitamente mencionadas en puertas entrantes del tipo de proceso o tipo de servicio. En el caso de conjunto de procesos o servicio, el <valid input signal set> no necesita mencionar señales que estén explícitamente mencionadas en rutas de señales que conducen al conjunto de procesos o servicio.

- El <valid input signal set> no necesita mencionar procedimientos remotos o variables remotas.

Si un procedimiento remoto o variable remota no se menciona en ninguna ruta de señales saliente de un conjunto de procesos o servicio, o en ninguna puerta saliente de un tipo de proceso o tipo de servicio, debe entonces mencionarse en una <imported procedure specification> o <imported variable specification> en aquel conjunto de procesos, tipo de proceso o tipo de servicio."

**Subcláusula 2.5.2**

*Gramática gráfica concreta:* en la regla de gramática para <signal route definition area>, "<signal route name>" es sustituido por "[ <signal route name>]", y la primera ocurrencia de "<signal list area>" es sustituida por "[<signal list area>]".

En el segundo párrafo después de las reglas de gramática, sustituir la primera ocurrencia de una "<signal list area>" por "al menos una <signal list area>".

**Subcláusula 2.5.2**

*Semántica:* en el primer párrafo, sustituir la primera frase por:

"Una *Signal-route-definition* representa una ruta de transporte de señales (que incluyen las señales implícitas supuestas a partir del intercambio de procedimiento remotos y variables remotas, véanse 4.13 y 4.14)."

Sustituir el penúltimo párrafo por:

"Cuando una instancia de señal es enviada a una instancia del mismo conjunto de instancias de proceso, la interpretación de *Output-node* implica, bien que la señal se pone directamente en el puerto de entrada del proceso de destino, o bien que la señal es enviada vía una ruta de señales que conecta el conjunto de instancias de proceso así mismo."

Añadir el siguiente párrafo:

"Un procedimiento remoto o variable remota en una ruta de señales se menciona como saliente de un importador y entrante a un exportador."

**Subcláusula 2.5.2**

*Modelo:* sustituir el texto completo por:

"*Implicit signal-route names:* Si el <signal route name> se omite desde una <signal route definition> o <signal route definition area>, la ruta de señales está nombrada implícita y unívocamente.

*Implicit signal routes and implicit signal lists on signal routes:* Si un bloque, tipo de bloque, conjunto de procesos (que contienen servicios) o tipo de proceso (que contiene servicios) no contienen rutas de señales explícitas, se derivan rutas de señales implícitas sin listas de señales. Después, las rutas de señales sin listas de señales explícitas se rellenan con señales, procedimientos remotos y variables remotas. Los detalles se describen en 7.3.3 y 7.3.5.

Si un bloque, tipo de bloque, conjunto de procesos (que contienen servicios) o tipo de proceso (que contiene servicios) contienen rutas de señales explícitas para señales, pero no contienen rutas de señales explícitas para procedimientos remotos y variables remotas, se derivan rutas de señales implícitas capaces de transportar los procedimientos remotos y las variables remotas en cuestión. Los detalles se describen en 7.3.4.

Los <signal route path> explícitamente definidos deben tener listas de señales no vacías después de estas transformaciones."

**Subcláusula 2.5.3**

*Gramática textual concreta:* añádase el siguiente texto inmediatamente después de las reglas de gramática:

"Ningún canal o ruta de señales puede mencionarse más de una vez en una <channel to route connection> o <signal route to route connection>. Ninguna ruta de señales puede mencionarse después de la palabra clave **and** en más de una <channel to route connection> o <signal route to route connection> de una unidad de ámbito dada.

Dos <channel to route connection> o <signal route to route connection> de una unidad de ámbito dada debe -antes de la palabra clave **and**- bien mencionar el mismo conjunto de canales/rutas de señales, o bien no tener canales/rutas de señales en común."

**Subcláusula 2.5.3**

*Gramática gráfica concreta:* suprimir el último párrafo.

Añadir el siguiente párrafo al final de la subcláusula:

"NOTA – Debido al constructivo **connect**, un canal o ruta (externa) de señales que puede ser anónimo en la versión gráfica de una especificación, puede precisar un nombre en la versión textual correspondiente. Esto es totalmente análogo al caso de <merge area> en gráficos de proceso, servicio o procedimiento.

Una herramienta que convierte la versión gráfica de una especificación a una versión textual, debe de este modo ser capaz de generar nombres de canales y rutas de señales implícitos."

**Subcláusula 2.5.3**

Se añade una nueva subcláusula denominada *Modelo* que contiene el texto siguiente:

"Si más de una <channel to route connection > o <signal route to route connection> de una unidad de ámbito menciona los mismos canales/rutas de señales antes de la palabra clave **and,** estas conexiones se unen."

**Subcláusula 3.2.2**

*Gramática textual concreta:* sustituir el penúltimo párrafo por:

"Si una <block substructure definition> contiene <channel definitions> (todas con <signal list> explícitas) y <textual typebased block definition>, cada puerta (que contiene solamente señales) de las <block type definition> de las <textual typebased block definition> debe estar conectada a un canal por lo menos.

Si una <block substructure definition> contiene <channel definitions> (alguna o todas con <signal list> implícitas o que mencionan procedimientos remotos o variables remotas) y <textual typebased block definition> cada puerta de las <block type definition> de las <textual typebased block definition> debe estar conectada a un canal por lo menos.

Las <channel connection> deben cumplir condiciones estáticas similares a las de las <channel to route connection> y <signal route to route connection> (véase 2.5.3 , *Gramática textual concreta).*"

Añadir el texto siguiente inmediatamente antes de *Gramática gráfica concreta:*

> "Un <channel path> explícito sin una lista de señales explícita puede tener un punto extremo en un bloque que tiene una versión no dividida y una versión dividida. En este caso, y después de que los <channel path> y los <signal route path> implícitos sin <signal list> han sido derivados (7.3.3 y 7.3.4), y la <signal list> en el < channel path> puede rellenarse en (7.3.5) de dos modos diferentes:
>
> - Como parte de una secuencia (de <channel path> y <signal route path> sin <signal list> explícitos) que conduce dentro o fuera de una versión *no dividida.*
> - Como parte de una secuencia (de <channel path> y…) que conduce dentro o fuera de una versión *dividida.*
>
> La <signal list> debe contener las mismas señales, procedimientos remotos y variables remotas en ambos casos."

**Subcláusula 3.2.2**

*Modelo:* añadir el siguiente párrafo inmediatamente después del párrafo existente:

> "Si más de una <channel connection> de una subestructura de bloque menciona los mismos canales antes de la palabra clave **and**, estas <channel connection> se fusionan."

**Subcláusula 3.2.3**

*Gramática textual concreta:* sustituir el párrafo inmediatamente después de la regla de gramática para <channel substructure definition> por:

> "El <channel substructure name> después de la palabra clave **substructure** solamente puede omitirse si la <channel definition> circundante tiene un <channel name> explícito. En este caso, el <channel substructure name> es el mismo que el <channel name>."

En el penúltimo párrafo: suprimir el texto, "y para cada uno debe haber exactamente una <channel endpoint conection>".

Insertar el siguiente texto entre los dos últimos párrafos:

> "Un canal puede contener un <channel path> sin una lista de señales explícita y tener una subestructura de canal asociada. En este caso, y después de que los <channel path> y <signal route path> implícitos sin <signal list> se han derivado (7.3.3 y 7.3.4), la <signal list> en el <channel path> puede rellenarse (7.3.5) de tres modos diferentes:
>
> - Como parte de una secuencia (de <channel path> y <signal route path> sin <signal list> explícitas) "puenteando" la subestructura de canal.
> - Como parte de una secuencia (de <channel path> y…) que conduce dentro de la subestructura de canal.
> - Como parte de una secuencia (de <channel path> y…) que conduce fuera de la subestructura de canal.
>
> La <signal list> debe ser la misma en los tres casos."

**Subcláusula 3.2.3**

*Gramática gráfica concreta:* en la regla de gramática para <channel substructure diagram>, sustituir "+" por "*".

Sustituir el párrafo inmediatamente después de esta regla de gramática por:

> "Una ocurrencia de <block identifier> o **env** identifica un punto extremo del canal fraccionado. La ocurrencia se coloca fuera del <frame symbol> próxima al punto extremo de alguno o todos los subcanales asociados en el <frame symbol>. Un <channel symbol> dentro del <frame symbol> y conectado a éste indica un subcanal."

**Subcláusula 3.2.3**

*Modelo:* en el párrafo b), sustituir la última frase por: "representan por una <channel connection> en la cual el <block identifier> o **env** ha sido sustituido por el nuevo canal apropiado."

**Subcláusula 3.3**

*Gramática textual concreta:* añadir el siguiente texto a la subcláusula:

> "Dentro de cada *gate-boundary-delimited part* (este término se explica en 7.3.4) de una especificación, todos los canales y todas las listas de señales en canales deben ser explícitos si la parte utiliza señales en diferentes niveles de refinamiento."

**Subcláusula 4.13**

*Gramática textual concreta:* sustituir el primer párrafo después de las reglas de gramática por:

> "El uso de **nodelay** se describe en 7.3.4 y 7.3.5."

Sustituir el cuarto párrafo después de las reglas de gramática por:

> "Una variable remota mencionada en una <import expression> debe existir en el conjunto completo de salidas (7.3.1) de un tipo de proceso, conjunto de procesos, tipo de servicio o servicio circundantes."

**Subcláusula 4.13**

*Modelo:* en el primer párrafo, sustituir las dos primeras frases por:

> "Una operación de importación se modela mediante intercambio de señales definidas implícitamente."

Después del tercer párrafo, añadir el párrafo siguiente:

> "En cada canal o ruta de señales que mencione la variable remota, esta variable remota es reemplazada por *xQUERY*. Para tal canal o ruta de señales, se añade un nuevo canal o ruta de señales en el sentido de flujo opuesto; este canal o ruta de señales transporta la señal *xREPLY*. En el caso de un canal, el nuevo canal tiene la misma característica de retardo que el canal original."

**Subcláusula 4.14**

*Gramática textual concreta:* sustituir el primer párrafo después de las reglas de gramática por:

> "El uso de **nodelay** se describe en 7.3.4 y 7.3.5."

Sustituir el cuarto párrafo después de las reglas de gramática por:

> "Un procedimiento remoto mencionado en una <remote procedure call> debe existir en el conjunto completo de salidas (7.3.1) de un tipo de proceso, conjunto de procesos, tipo de servicio o servicio circundantes."

**Subcláusula 4.14**

*Modelo:* en el primer párrafo sustituir las dos primeras frases por:

> "Una llamada a procedimiento remoto es modelada por intercambio de señales definidas implícitamente."

Después del segundo párrafo, añadir el siguiente párrafo:

> "En cada canal o ruta de señales que mencione el procedimiento remoto, este procedimiento remoto es reemplazado por *pCALL*. Para tal canal o ruta de señales, se añade un nuevo canal o ruta de señales en el sentido de flujo opuesto; este canal o ruta de señales transporta la señal *p*REPLY. En el caso de un canal, el nuevo canal tiene la misma característica de retardo que el canal original."

**Subcláusula 6.1.1.3**

*Semántica:* suprimir el segundo párrafo.

**Subcláusula 6.1.1.4**

*Semántica:* suprimir el segundo párrafo.

**Subcláusula 6.1.4**

*Gramática textual concreta:* en el tercer párrafo, línea 4, modificar el texto "si el conjunto de señales" para que diga "si el conjunto de señales (si están especificadas)".

Sustituir los párrafos cuarto y quinto por:

> "Si el tipo denotado por <base type> en una <textual typebased block definition> o <textual typebased process definition> contiene rutas de señales, se aplica la siguiente regla: Para cada combinación de (puerta, señal, sentido de flujo) definida por el tipo, el tipo debe contener al menos una ruta de señales que -para el sentido de flujo dado- mencione **env** y la puerta y mencione la señal o no tenga una <signal list> explícita asociada. En el último caso, debe ser posible derivar que la ruta de señales es capaz de transportar la señal en el sentido de flujo dado.
>
> Si el tipo de bloque contiene señales de ruta que mencionan procedimientos remotos o variables remotas, se aplica una regla similar.
>
> Si una <block substructure definition> en el tipo de bloque denotado por <base type> en una <textual typebased block definition> contiene canales, se aplica la siguiente regla: Para cada combinación de (puerta, señal, sentido de flujo) definida por el tipo de bloque, el tipo debe contener al menos un canal, que -para el sentido de flujo dado- mencione **env** y la puerta y mencione la señal o no tenga <signal list> explícita asociada. En el último caso, debe poderse derivar que canal es capaz de transportar la señal en el sentido de flujo dado.
>
> Si el tipo de bloque contiene canales que mencionan procedimientos remotos o variables remotas, se aplica una regla similar.
>
> Se pueden utilizar trayectos de comunicación explícitos para señales "normales", pero trayectos de comunicación implícitos para procedimientos remotos y variables remotas, incluso en el caso en que la especificación define explícitamente y/o utiliza puertas que mencionan procedimientos remotos o variables remotas.
>
> Se puede especializar la puerta implícita rpv_gate explícitamente en un subtipo. Una especialización explícita de rpv_gate es tratada como una especialización de cualquier otra puerta (por ejemplo, está permitido añadir señales "normales" a rpv_gate).
>
> En un subtipo de un tipo con trayectos de comunicación explícitos para señales, pero con trayectos de comunicación implícitos para procedimientos remotos y variables remotas, es posible añadir <channel path> o <signal route path> explícitos que no contengan <signal list> explícitas, o que mencionen procedimientos remotos o variables remotas."

**Subcláusula 6.1.4**

*Modelo:* añadir al principio el siguiente texto:

> "*Puertas implícitas para procedimientos remotos y variables remotas:* una puerta implícita llamada rpv_gate se deriva para cada tipo de bloque, tipo de proceso y tipo de servicio que (directamente o a través de instancias contenidas) exporta procedimientos remotos o variables remotas, y que no tiene ninguna puerta explícita que mencione estos procedimientos remotos o variables remotas. Los detalles se describen en la 7.3.2.
>
> Después de esta transformación, cada procedimiento remoto o variable remota exportado o importado debe ser mencionado por lo menos en una puerta (en el sentido de flujo apropiado) del tipo."

Al principio del primer párrafo del texto existente, añadir el título "*Transformación de puertas:"*

**Cláusula 7**

Añadir la siguiente subcláusula:

## 7.3 Inserción de puertas, canales, rutas de señales y listas de señales implícitos

Se aplica la transformación a continuación en el orden dado.

### 7.3.1 Conjuntos completos de entradas y salidas

*<valid input signal set> completos:* el <valid input signal set> completo de una <process type definition>, <process definition>, <service type definition> o <service definition> es el conjunto de señales, procedimientos remotos y variables remotas que:

- ocurre en el <valid input signal set> completo del supertipo, en su caso, del tipo de proceso o servicio (en el caso de una <process type definition> o <service type definition>); o
- son mencionados en las puertas de entrada del tipo de proceso o servicio (en el caso de una <process type definition> o <service type definition>), o son mencionados explícitamente en rutas de señales que conducen al conjunto de procesos o servicio (en el caso de una <process definition> o <service definition>); o
- son exportados por las definiciones de procedimientos o definiciones de variables en el tipo de proceso, conjunto de procesos, tipo de servicio o servicio (solamente para procedimientos remotos y variables remotas).

En el caso de una <process type definition> o <process definition> que contiene servicios, el <valid input signal set> completo contiene además las señales, procedimientos remotos y variables remotas que están contenidas en los <valid input signal set> completos de estos servicios.

*Conjuntos completos de entradas y salidas:* El conjunto completo de entradas de una <process type definition>, <process definition>, <service type definition> o <service definition> es su <valid input signal set> completo. El conjunto completo de salidas de <process type definition>, <process definition>, <service type definition> o <service definition> es el conjunto de señales, procedimientos remotos y variables remotas que:

- ocurre en el conjunto completo de salidas del supertipo, en su caso, del tipo de proceso o servicio (en el caso de una <process type definition> o <service type definition>), o
- son mencionados en las puertas salientes del tipo de proceso o tipo de servicio (en el caso de una <process type definition> o <service type definition>), o son mencionados explícitamente en rutas de señales que conducen desde el conjunto de procesos o servicio (en el caso de una <process definition> o <service definition>), o
- son mencionados en los nodos salientes del conjunto de procesos o servicio (en el caso de una <process definition> o <service definition>) (solamente para señales), o
- son mencionados en las especificaciones de procedimiento importado y las especificaciones de variable importada del tipo de proceso, conjuntos de procesos, tipo de servicio o servicio (solamente para procedimientos remotos y variables remotas).

En el caso de una <process type definition> o <process definition> que contiene servicios, el conjunto completo de salidas contiene además las señales, procedimientos remotos y variables remotas que están contenidos en los conjuntos completos de salidas de estos servicios.

NOTA – Los conjuntos completos de entradas y salidas que aquí se describen no incluyen señales de temporizador o señales definidas implícitamente."

### 7.3.2 Puertas implícitas

*Puertas implícitas para procedimientos remotos y variables remotas:* Una <gate definition> implícita se añade a cada tipo de bloque, tipo de proceso o tipo de servicio que:

- directamente o a través de instancias contenidas, exporta o importa procedimientos remotos o variables remotas, y
- no tiene puertas explícitas que mencionen estos procedimientos remotos o variables remotas.

La puerta denotada por esta <gate definition> tiene el nombre rpv_gate, tiene una parte entrante si el tipo exporta al menos un procedimiento remoto o variable remota, y tiene una parte saliente si el tipo importa al menos un procedimiento remoto o variable remota. Cada parte menciona los procedimientos remotos o variables remotas que son comunicados en el sentido de flujo correspondiente.

La puerta implícita rpv_gate (si está presente) y los trayectos de comunicación implícitos especialmente para procedimientos remotos o variables remotas, se añaden al supertipo antes de que el contenido del supertipo y del subtipo se fusionen.

Si el tipo hereda tal puerta implícita a partir de otro tipo y exporta o importa nuevos procedimientos remotos o variables remotas, la nueva <gate definition> implícita contiene la palabra clave **adding** y menciona los nuevos procedimientos remotos y variables remotas.

### 7.3.3 Canales y rutas de señales implícitos

En la descripción a continuación se entiende que cuando se derivan canales o rutas de señales implícitos, los constructivos **connect** se derivan también en los casos en que es apropiado.

*Canales implícitos:* Si una <system definition> o <system type definition> contiene <channel definition>, se derivan <channel definition> implícitas sin <signal list> de manera que cada posible pareja de "puntos extremos de canal" son conectados en cada sentido de flujo posible. Un "punto extremo de canal" tiene aquí uno de los siguientes significados:

- Una <block definition>.
- Una puerta de una <textual typebased block definition>.
- El entorno del sistema.

Sin embargo, una <channel definition> implícita no conecta una <block definition> consigo misma en su lugar, un bloque denotado por una <block definition> comunica consigo vía trayectos de comunicación implícitos *internos.*

Además, si una <block substructure definition> (de una <block definition> o <block type definition>) o <channel substructure definition> no contiene <channel definition>, se derivan <channel definition> implícitas de una manera similar. Aquí, un "punto extremo de canal" tiene el mismo significado que anteriormente, o uno de los siguientes:

- Un canal (exterior) conectado a la unidad de ámbito circundante (en el caso de una <block definition>).
- Una puerta de la unidad de ámbito circundante (en el caso de una <block type definition>).
- El canal "circundante" (en el caso de una <channel substructure definition>).

Las subestructuras anidadas se tratan después recursivamente de la misma forma.

Todos los canales implícitos son unidireccionales y con retardo.

*Rutas de señales implícitas:* Si una <block definition> o <block type definition> no contiene <signal route definition> explícitas, se derivan <signal route definition> implícitas sin <signal list> tales que cada pareja posible de "puntos extremos de ruta de señales" están conectados en cada sentido de flujo posible. Un "punto extremo de ruta de señales" tiene aquí uno de los siguientes significados:

- Una < process definition>.
- Una puerta de una <textual typebased process definition>.
- Un canal conectado a la unidad de ámbito circundante (en el caso de una <block definition>).
- Una puerta de la unidad de ámbito circundante (en el caso de una <block type definition>).
- El entorno del sistema (en el caso de una <block definition> definida como un sistema).

Sin embargo, ninguna <signal route definition> implícita conecta una <process definition> consigo misma; en su lugar, un conjunto de procesos denotado por una <process definition> comunica consigo misma vía trayectos de comunicación *internos* (si están descompuestos en servicios) o sin ningún trayecto de comunicación (si no están descompuestos en servicios).

Todas las rutas de señales implícitas son unidireccionales.

Además, si una <process definition> o <process type definition> contiene servicios, pero no contiene rutas de señales, se derivan <signal route definition> de una manera similar.

*Canales y rutas de señales implícitos para procedimientos remotos:* Si una unidad de ámbito contiene canales o rutas de señales explícitos para señales, pero no contiene canales o rutas de señales explícitos para procedimientos remotos y variables remotas, se derivan canales para rutas de señales implícitos para procedimientos remotos y variables remotas solamente, en la unidad de ámbito en cuestión. En 7.3.4 se describen las condiciones precisas acerca de cuándo y cómo esto tiene lugar.

### 7.3.4 Canales y rutas de señales implícitos para procedimientos remotos y variables remotas

En lo que viene a continuación, conviene utilizar el término *gate-boundary* (frontera de puerta). Una frontera de puerta es la frontera de una <system type definition> <block type definition>, <process type definition> o <service type definition>.

Una especificación SDL se compone de este modo de partes delimitadas por fronteras de puerta. En lo que viene a continuación estas partes se denominan partes *gate-boundary-delimited*.

Dentro de cada parte delimitada por fronteras de puerta, se derivan canales o rutas de señales implícitos especialmente para procedimientos remotos o variables remotas si se cumplen las siguientes condiciones (es decir, dentro de cada parte delimitada por fronteras de puerta, los procedimientos remotos o variables remotas se tratan del mismo modo que las señales "normales" salvo que se cumplan las siguientes condiciones):

- Todos los <channel path> y <signal route path> (puede no haber ninguno) en la parte delimitada por fronteras de puerta contienen <signal list> explícitas.
- Ninguna <signal list> en los <channel path> o <signal route path> en la parte menciona un procedimiento remoto o variables remota.

En este caso, cualesquiera canales y rutas de señales implícitos en las unidades de ámbito afectadas se derivan de manera similar a la descrita en 7.3.3 y 7.3.5, pero con la siguiente modificación: Cada <channel path> o <signal route path> implícito derivado de acuerdo con las reglas de 7.3.3, es reemplazado por dos o tres <channel path> o <signal route path>. La <signal list> del <channel path> o <signal route path> implícito "original" y derivado de acuerdo con las reglas de 7.3.5, se divide entre los dos o tres <channel path> o <signal route path> del modo siguiente:

- Solamente se rellena un trayecto con señales "normales". Si el trayecto es un <channel path>, se retarda. Este trayecto *no* se deriva si la unidad de ámbito continente ya contiene canales o rutas de señales para señales "normales".
- Solamente se rellena un trayecto con procedimientos remotos o variables remotas que han sido definidas *sin* el atributo **nodelay**. Si el trayecto es un <channel path>, se retarda.
- Solamente se rellena un trayecto con procedimientos remotos y variables remotas que han sido definidos *con* el atributo **nodelay**. Si el trayecto es un <channel path>, *no* se retarda.

### 7.3.5 Listas de señales implícitas en canales y rutas de señales

*Listas de señales implícitas en canales y rutas de señales:* La especificación contiene ahora algunas secuencias de <channel path> y <signal route path> donde cada secuencia tiene las siguientes propiedades:

- Cada <channel path> o <signal route path> (salvo el último) en la secuencia es conectado al siguiente <channel path> o < signal route path>.
- Ninguno de los <channel path> o < signal route path> en la secuencia contiene <signal list>.
- El "punto extremo" de origen del primer <channel path> o <signal route path> es uno los siguientes:
  - Un número de <channel path> o < signal route path>, alguno de los cuales tienen <signal list> explícitas.
  - Una puerta.
  - Una <process definition> que no contiene servicios.
  - Una <service definition>.
  - El entorno del sistema.

El "punto extremo" de destino del último <channel path> o <signal route path> satisface una condición similar.

Las <signal list> en los <channel path> y <signal route path> en las secuencias se rellenan ahora de modo que puedan transportar "tantos" tipos de señales, llamadas a procedimiento remoto y peticiones de importación como sea posible. Esta derivación utiliza los conjuntos completos de entradas y salidas de cualesquiera <process definition> y <service definition> en los "puntos extremos" de las secuencias.

Esto es, cada secuencia contribuye con la intersección de los siguientes conjuntos de señales, procedimientos remotos y variables remotas, a todas las <signal list> en la secuencia:

- El conjunto de origen de la secuencia (del que se hace caso omiso si el "punto extremo" de origen es el entorno del sistema).
- El conjunto de destino de la secuencia (del que se hace caso omiso si el "punto extremo" de destino es el entorno del sistema).
- El conjunto de señales, procedimientos remotos y variables remotas que son visibles a lo largo de la secuencia completa.

Cuando se rellena un procedimiento remoto o variable remota de un canal dado, la propiedad delay/**nodelay** del procedimiento remoto o variable remota se ignora a menos que el canal haya sido derivado de acuerdo con las reglas de 7.3.4. Si un <channel path> o < signal route path> se ha derivado de acuerdo con las reglas en 7.3.4, la <signal list> en este trayecto se rellena de acuerdo con las reglas adicionales en 7.3.4.

En este momento se eliminan todos los <channel path> y <signal route path> implícitos con listas de señales vacías. Los constructivos **connect** que hacen "huérfano" este camino, son también eliminados".

# 10 Uso de un bloque, un proceso o un servicio como un sistema

La visión SDL de un sistema SDL es una entidad autocontenida que comunica con su entorno mediante señales. Sin embargo, a menudo existe más de una visión. Véase por ejemplo un ordenador. Puede contemplarse como un sistema bien definido en sí mismo, pero puede también ser un componente de un sistema mayor (por ejemplo una red de ordenadores). Es útil por tanto permitir que un servicio, un proceso o un bloque sea considerado un sistema en sí mismo. Esta característica es también útil para sistemas sencillos que se componen, por ejemplo, de un proceso.

Una <system definition> puede de este modo estar constituida por solamente un bloque (posible type-based), proceso o servicio. Obsérvese que una ampliación separada descrita en otra parte permite a una unidad de ámbito tener una <use clause>, lo que significa que tal <system definition> puede todavía basarse en entidades definidas en un paquete.

Las conexiones de ruta de señales/canal son omitidas para el nivel de proceso/bloque más alejado.

## 10.1 Cambios

**Subcláusula 2.4.1.1**

*Gramática concreta*, añadir: la <package list> y la <system definition> no pueden omitirse. La <package list> solamente puede especificarse si la <system definition> es una <textual system definition> o un <system diagram>.

**Subcláusula 2.4.1.3**

Cambiar la producción de <system definition> para que diga:

<system definition> ::=

<textual system definition>

| <system diagram>

| <block definition>

| <block diagram>

| <textual typebased block definition>

| <graphical typebased block definition>

| <process definition>

| <process diagram>

| <textual typebased process definition>

| <graphical typebased process definition>

| <service diagram>

| <textual typebased service definition>

| <graphical typebased service definition>

**Subcláusula 2.4.1.3**

Añadir a *Modelo:*

1) Si una <system definition> es una <service definition> o una <textual typebased service definition>, es sintaxis derivada para una <process definition> que tiene el mismo nombre que el servicio, contiene rutas de señales implícitas y contiene la <service definition> o <textual typebased service definition> como la única definición.

   Si una <system definition> es un <service diagram> o una <graphical typebased service definition>, es sintaxis derivada para un <process diagram> que tiene el mismo nombre que el servicio, contiene rutas de señales implícitas y contiene el <service diagram> o <graphical typebased service definition> como la única definición.

2) Si una <system definition> es una <process definition> o una <textual typebased process definition>, es sintaxis derivada para una <block definition> que tiene el mismo nombre que el proceso, contiene rutas de señales implícitas y contiene la <process definition> o <textual typebased process definition> como la única definición.

   Si una <system definition> es un <process diagram> o una <graphical typebased process definition>, es sintaxis derivada para un <block diagram> que tiene el mismo nombre que el proceso, contiene rutas de señales implícitas y contiene el <process diagram> o <graphical typebased process definition> como la única definición.

3**)** Si una <system definition> es una <block definition> o una <graphical typebased block definition>, es sintaxis derivada para una <system definition> que tiene el mismo nombre que el bloque, contiene canales implícitos y contiene la <block definition> o <textual typebased block definition> como la única definición.

   Si una <system definition> es un <block diagram> o una <textual typebased block definition>, es sintaxis derivada para un <system diagram> que tiene el mismo nombre que el bloque, contiene canales implícitos y contiene el <block diagram> o <graphical typebased block definition> como la única definición.

Para las dos condiciones a continuación los contenidos de subestructura contenidos en una <block diagram> o una <block definition> se consideran directamente contenidos en el <block diagram> o la <block definition>:

Una <channel definition area> o una <signal route definition area> para un canal o ruta de señales directamente contenidos en una <system definition> no puede ser *asociada con* (*associated with)* <channel identifiers> o <external signal route identifiers>.

La unidad de ámbito directamente contenida en una <system definition> no puede contener <channel to route connection>, <channel connection> o <signal route to route connection>.

# 11 Expresión de estado

## 11.1 Motivo

Junto con el uso de estado asterisco, es necesario a menudo tener acceso a información acerca de la identidad del estado más recientemente introducido. Tal característica forma parte del SDL-96.

## 11.2 Solución

Se he introducido un nuevo operador imperativo: **estado**. Denota el valor de la cadena de caracteres que contiene la ortografía del estado más recientemente introducido.

### 11.3 Ejemplo

**state** *; /* Say covering among others the state named state1 */

**input** *;

...

**decision state**;

(state1) : **task** 'do something for state1';

**else** : **task** 'do something else';

**enddecision**;

### 11.4 Cambios

**Subcláusula 5.4.4**

Añadir una alternativa en <imperative operator>

<imperative operator> ::=

<now expression>

| ...

| <anyvalue expression>

| <state expression>

Añadir un nueva **subcláusula 5.4.4.7**:

*Gramática textual concreta*

<state expression> ::=

**state**

*Modelo*

la <state expression> es sintaxis derivada para un literal cadena de caracteres que contiene la ortografía en letra pequeña del nombre del estado más recientemente introducido de la unidad de ámbito circundante más próxima. Si no existe tal estado, <state expression> denota la cadena vacía ('). El constructivo se transforma junto con <dash nextstate> (véase 7.1 paso 18).

## 12 Uso ampliado de paquetes

### 12.1 Motivo

Actualmente los paquetes solamente pueden utilizarse en paquetes y en el nivel de sistema. El inconveniente de esto es que todas las definiciones en un paquete serán visibles en el ámbito de sistema completo. A menudo es necesario incluir definiciones en un nivel inferior. Por ejemplo, es frecuente el caso en el cual algún tipo de datos solamente se necesita en un solo proceso en el sistema, pero ese tipo de datos es de todos modos tan general que puede ser reutilizado en otros sistemas. Por consiguiente, es necesario poder utilizar paquetes en la unidad de ámbito en que se precise. Tal cambio hará también más fácil la integración de SDL y GDMO, ya que GDMO permite el uso de paquetes en el nivel de clase de objeto manejado (que correspondería a nivel de tipo de proceso SDL).

### 12.2 Solución

Permitir el uso de paquetes en otras unidades de ámbito distintas del sistema, tales como tipos bloque/tipos de bloque, proceso/tipos de proceso, servicio/tipo de servicio, etc.

### 12.3 Cambios

En la página 18, el párrafo que comienza con "Se puede permitir ...": después de la frase que termina con "es el mismo que la parte más a la derecha del <qualifier> completo que denota esa unidad de ámbito", añadir una nueva frase:

> En relación a la visibilidad y el uso de calificadores una <package reference clause> asociada a una unidad de ámbito S se considera que representa una definición de paquete directamente circundante S y definida en la unidad de ámbito en la que se define S.
>
> NOTA – En la sintaxis concreta, los paquetes no pueden definirse dentro de otras unidades de ámbito. La regla anterior se utiliza solamente para definir las reglas de visibilidad que se aplican a los paquetes. Una consecuencia de esta regla es que se puede hacer referencia a nombres en un paquete utilizando calificadores diferentes, uno para cada <package reference clause> envuelta del paquete.

En la página, 24, tercera frase sustituir: "Las definiciones dentro de un paquete son visibles para un sistema u otros paquetes" por: "Las definiciones dentro de un paquete son visibles para otra unidad de ámbito."

En la página 25, en la última frase entre "1)" y "2)" sustituir: "otro <package> o la <system definition> " por " una unidad de ámbito"

En la página 25, apartado 1), sustituir: "El package> o la <system definition> tiene una <package reference clause> que menciona el <package> y" por:

> "La unidad de ámbito tiene una <package reference clause> asociada, o alguna unidad de ámbito que incluye la unidad de ámbito tiene una <package reference clause> asociada y la <package reference clause> menciona el <package> y"

página 26: suprimir el párrafo que comienza con "Los nombres que tienen su ocurrencia de definición en un <package> están referenciados ...."

página 26, *Modelo* añadir al final:

> NOTA – Cuando un tipo se especializa (véase 6.3) ninguna <package reference clause> asociada al supertipo es copiada al tipo especializado.

#### 12.3.1 Cambios generales a la gramática textual concreta

Añadir "{<package reference clause>}*" al principio del lado derecho en las definiciones de <block definition>, <process definition>, <service definition>, <procedure definition>, <operator definition>, <block substructure definition>, <channel substructure definition>, <system type definition>, <block type definition>, <process type definition>, y <service type definition>.

#### 12.3.2 Cambios generales a la gramática gráfica concreta

Añadir: "{<package reference area > *está asociada con* (*is associated with*)" al principio del lado derecho en las definiciones de <block diagram>, <process diagram>, <service diagram>, <procedure diagram>, <operator diagram>, <block substructure diagram>, <channel substructure diagram>, <system type diagram>, <block type diagram>, <process type diagram>, y <service type diagram>.

Añadir una frase: "La <package reference area> debe colocarse en la parte superior del símbolo de recuadro del sistema." en las subcláusulas en las que se definen los diagramas mencionados anteriormente.

## 13 Operadores con argumentos cero en la parte ***operadores*** de un neotipo

### 13.1 Motivo

El propósito original de los operadores con argumentos cero (es decir, literales) fue probablemente utilizarlos para contribuir al conjunto de valores de un género (por ejemplo, para definir "valores enumerados"). La facilidad **ordering** indica esto.

En SDL-92 hay la posibilidad de utilizar en su lugar sinónimos, pero esto acarrea los siguientes inconvenientes:

- Los sinónimos no pueden ser sobrecargados.
- Las "signaturas" de los sinónimos se definen en un lugar de la especificación distinto que las signaturas de los correspondientes operadores "non-nullary" (el **newtype** "exterior" e "interior", respectivamente). Si el **newtype** tiene parámetros de contexto, es realmente imposible definir el sinónimo exterior al **newtype.**
- El "comportamiento" de los sinónimos se define en un lugar de la especificación distinto del comportamiento de los correspondientes operadores "non-nullary". En primer lugar, los sinónimos se definen "fuera" del **newtype** mientras que las definiciones de los axiomas u operadores algorítmicos se dan "dentro" del **newtype**. En segundo lugar (en el caso de operadores algorítmicos), el comportamiento de los operadores "non-nullary" puede definirse en diagramas de operadores **referenced** -es decir, una vez separado del "comportamiento" de los correspondientes sinónimos.
- No pueden utilizarse sinónimos en los axiomas.

## 13.2 Solución

En SDL-96 están por tanto permitidos operadores con argumento cero en la parte **operators** de un **newtype**. En oposición a los operadores "nullary" en la parte **literals**, los operadores "nullary" en la parte **operators** no están influenciados por facilidades como **ordering**. Por el contrario, se dispone de las mismas facilidades para los operadores "nullary" en la parte **operators** al igual que para los demás operadores -por ejemplo, la posibilidad de definirlos algorítmicamente o externamente y la posibilidad de definir operadores "nullary" "nuevos" de tipos "viejos" (por ejemplo, Pi de tipo Real).

## 13.3 Ejemplo

```
newtype Color
    literals Yellow, Blue, Green, Red;
    operators
        ordering;
        First: -> Color; /* not allowed  SDL-92  */
        Last : -> Color; /* not allowed  SDL-92 */
        /* other operators */
    axioms
        First == Yellow;
        Last  == Red;
        /* other axioms */
endnewtype;
```

los literales Amarillo, Azul, Verde y Rojo constituyen el conjunto de valores de Color, y los cuatro literales son los ordenados por **ordering**. Actualmente, Primero y Último se definen también bajo **literals** y la ordenación debe definirse entonces de una manera mucho más complicada (para el usuario) que declarando solamente **ordering** bajo **operators**.

## 13.4 Repercusión en la Recomendación Z.105

La ampliación hace posible un acceso mucho más amigable para el usuario a los operadores Primero y Último definidos para el tipo de enumeración predefinida de la Recomendación Z.105. Como en SDL-92 no están permitidos operadores con argumentos cero, cuando se utilicen aquellos dos operadores debe dárseles un argumento "ficticio".

Para la versión Z.105 que corresponde a SDL-96, los dos operadores están sobrecargados:

```
First: -> Enumeration;
Last: -> Enumeration;
```

## 13.5 Cambios

**Subcláusula 5.2.2**

Cambiar el título para que diga "**Literales y operadores**".

**Subcláusula 5.2.2**

*Gramática abstracta:* las reglas de gramática para *Signature, Literal-signature, Operator-signature* y *Argument-list* se sustituyen por:

*Signature :: Operator-name*

*Argument-list*

*Result*

*Argument-list = Sort-reference-identifier**

Se suprime la regla de gramática para *Literal-operator-name*.

**Subcláusula 5.2.2**

*Gramática textual concreta:* la regla de gramática para <literal signature> se sustituye por:

<literal signature> ::=

<literal name> |

<name class literal>

<literal name> ::=

<literal operator name> |

<extended literal name>

Se cambia la regla de gramática <argument list> por:

<argument list> ::=

[ <argument sort> { , <argument sort> }* ]

Se sustituye el texto completo después de las reglas de gramática por:

"Las alternativas <name class literal>, <extended literal name>, <ordering>, <noequality>, <extended operator name>, <generator sort> y <syntype> no forman parte del núcleo de datos.

Una *Signature* se representa por una <literal signature> que contiene un <literal name> que no es un <generator formal name>, o por una <operator signature> que contiene un <operator name> que no es un <generator formal name>.

Si la *Signature* se representa por una <literal signature>, la *Argument-list* está vacía, y el *Result* es el género introducido por la <partial type definition> que define la <literal signature>. Si la *Signature* se representa por una <operator signature>, cada *Sort-reference-identifier* de la *Argument-list* es representado por un <argument sort> de la <argument list>, y el *Result* se representa por el <result>.

Las otras formas posible de <literal signature> y <operator signature> son notaciones taquigráficas. Cada una de estas notaciones taquigráficas corresponde en general a varias *Signatures.*

Un <literal name> u <operator name> que no es un <generator formal name> corresponde a un *Operator-name*. Este *Operator-name* es único dentro de la unidad de ámbito definitoria, incluso aunque el <literal name> u <operator name> correspondiente pueda no ser único.

El *Operator-name* único se obtiene a partir de:

1) el <literal name> u <operator name>; más
2) la lista (posiblemente vacía) de identificadores de género de argumento; más
3) el identificador de género de resultado; más
4) el identificador de género de la definición de tipo parcial en que se define el <literal name> u <operator name>.

Siempre que se especifica un <literal identifier> u <operator identifier>, el *Operator-name* único en *Operator-identifier* se deriva del mismo modo con la lista de géneros de argumento y el género de resultado derivado del contexto. Dos operadores (cada uno de los cuales puede ser un literal o un operador "normal") con el mismo nombre pero que difieren en uno o más de los géneros de argumento o resultado, tienen *Operator-names* diferentes.

Siempre que un <qualifier> de un <literal identifier> u <operator identifier> contiene un <path item> con la palabra clave **type**, el <sort name> después de esta palabra clave no forma parte del *Qualifier* del *Operator-identifier*, pero se utiliza para derivar el *Name* único de este *Identifier*. En este caso el *Qualifier* se forma a partir de la lista de <path item> que precede a la palabra clave **type**."

**Subcláusula 5.2.2**

*Semántica:* se cambian los párrafos tercero y cuarto para que digan:

"Un operador (sin argumentos) definido en una <literal signature> se denomina un literal. Un operador sin argumentos y definido en una <operator signature> se denomina un operador "nullary".

Un literal o un operador "nullary" representan un valor fijo que pertenece al género resultado del operador."

Se inserta el siguiente nuevo párrafo inmediatamente antes del último:

"NOTA 1 – Desde el punto de vista de la gramática abstracta, no hay diferencia entre literales y operadores "nullary". La diferencia se encuentra en cómo se utilizan los dos en SDL: ciertas facilidades, por ejemplo, **ordering**, se aplican solamente a literales, no a operadores "nullary". Los operadores "nullary" se tratan como operadores con uno o más argumentos -por ejemplo, su comportamiento puede definirse mediante <operator definition>."

Cambiar la nota existente en la versión 1993 por:

"NOTA 2 – Directriz: una <operator signature> debe mencionar *normalmente* el género introducido por la <partial type definition> circundante bien como un <argument sort> o como un <result>. La excepción a esta directriz es el caso en que una <partial type definition> se utiliza exactamente como un "contenedor" ("minipaquete") para agrupar juntos operadores "nuevos" en géneros "viejos" cuando estos operadores están relacionados de algún modo. El <sort name> definido por tal <partial type definition> debe utilizarse solamente en <qualifier> de los operadores definidos."

Añadir al final el siguiente nuevo párrafo:

"NOTA 3 – directriz: un operador (sin argumentos) que se utiliza para contribuir al conjunto de valores de un género, debe definirse como un literal. Un operador (sin argumentos) que se utiliza para definir un nombre de un valor "existente" debe definirse como un operador "nullary". Por ejemplo, esta directriz ha sido utilizada para definir el género enumerado siguiente:

```
newtype Color

    literals

        Red, Green, Blue, Yellow;

    operators

        ordering;

        First: -> Color;

        Last : -> Color;

        /* other operators */

    axioms

        First == Red;

        Last  == Yellow;

        /* other axioms */

endnewtype Color;
```

"

**Subcláusula 5.2.3**

*Gramática abstracta:* cambiar la regla de gramática para *Ground-term* de forma que diga:

*Ground-term :: Operator-identifier Ground-term** |

*Conditional-ground-term*

Se suprime la regla de gramática para *Literal-operator-identifier*.

**Subcláusula 5.2.3**

*Gramática textual concreta:* cambiar la regla de gramática para <ground term> para que diga:

<ground term> ::=

<literal operator identifier> |

<operator identifier> ( [ <ground term> ] { , <ground term> }* ) |

( <ground term> ) |

<extended ground term>

En el párrafo que comienza con "c)", sustituir "*literal operator identifier*" por "*Operator-identifier*", y "literal" por "literal u operador".

En el cuarto párrafo contado desde el final, cambiar "<operator name>" por "<literal name> u <operator name>" (aparece dos veces) y "operador" por "literal u operador" (aparece tres veces).

En el penúltimo párrafo, cambiar "*operator identifier* o *literal operator identifier*" por "*Operator-identifier*".

Cambiar el último párrafo para que diga:

"NOTA – directriz: un axioma debe ser pertinente para los literales y operadores de la definición de tipo parcial circundante al mencionar un literal u operador cuya signatura se define dentro de la definición de tipo parcial. Un axioma dado debe ocurrir una sola vez."

**Subcláusula 5.3.1**

*Gramática textual concreta:* cambiar la regla de gramática para <extended ground term> para que diga:

<extended ground term> ::=

<extended literal identifier> |

<extended operator identifier> ( [ <ground term> ] { , <ground term> }* ) |

<ground term> <infix operator> <ground term> |

<monadic operator> <ground term> |

<conditional ground term>

Se suprime el <name class literal> alternativo de la letra de gramática para <extended literal name>.

**Subcláusula 5.3.1.2**

*Gramática textual concreta:* en el segundo párrafo después de las reglas de gramática, cambiar "*Literal-operator-identifier*" por "*Operator-identifier*". En el tercer párrafo, cambiar "*Literal-operator-name*" por "*Operator-name*".

**Subcláusula 5.3.1.7**

*Gramática abstracta:* en el último párrafo, se cambia "*literal operator identifier*" por "*Operator-identifier* correspondiente a un literal".

**Subcláusula 5.3.1.7**

*Modelo:* añadir el siguiente párrafo al final:

"NOTA – La palabra clave **ordering** no afecta a los operadores "nullary" definidos en <operator signature>."

**Subcláusula 5.3.1.11**

*Modelo:* añadir al final el siguiente párrafo:

"NOTA 2 – Los literales heredados se renombran mediante <literal renaming>. Los operadores "nullary" heredados son manejados por la palabra clave **all** o son renombrados por medio de <inheritance list>, exactamente igual que todos los demás operadores "normales" heredados."

(Nota del editor – la nota existente en la versión de 1993 se numera NOTA 1.)

**Subcláusula 5.3.1.15**

*Modelo:* añadir al final el siguiente párrafo:

"NOTA – Las correspondencias de literales no afectan a los operadores "nullary" definidos en <operator signature>."

**Subcláusula 5.3.2**

*Gramática textual concreta:* en la regla de gramática para <operator definition>, cambiar "<formal parameters> <end>" por "[<formal parameters> <end>]". En la regla de gramática para <textual operator reference>, cambiar "<formal parameters >" por "[<formal parameters>]".

En el tercer párrafo después de reglas de gramática, añadir el texto "(si está presente)" después de "<formal parameters>".

En el quinto párrafo, sustituir "<formal parameters>" por "[<formal parameters>]" (dos veces).

**Subcláusula 5.3.2**

*Gramática gráfica concreta:* en la regla de gramática para <operator heading>, sustituir "<formal parameters>" por "[<formal parameters>]".

**Subcláusula 5.3.2**

*Modelo:* añadir el siguiente párrafo al final:

"NOTA – No es posible especificar una <operator definition> para una <literal signature>."

**Subcláusula 5.3.3.2**

*Gramática textual concreta:* se modifica la regla de gramática para <ground primary> para que diga:

<ground primary> ::=

<literal identifier> |

<operator identifier> ( [ <ground expression list> ] ) |

( <ground expression> ) |

<conditional ground expression>

## 14 Reformulación de la subcláusula 6.3.2 concerniente a virtuales

Un tipo de bloque, tipo de proceso, tipo de servicio, o procedimiento puede especificarse como un tipo virtual cuando es definido localmente con otro tipo (denotado como el tipo *enclosing (circundante)* en esta subcláusula). Un tipo virtual puede ser redefinido en especializaciones del tipo circundante.

*Gramática textual concreta:*

<virtuality> ::=

**virtual** | **redefined** | **finalized**

<virtuality constraint> ::=

**atleast** <identifier>

<virtuality> y <virtuality constraint> son parte de una definición de tipo. Véanse 6.1.1.2 (tipo de bloque), 6.1.1.3 (tipo de proceso), 6.1.1.4 (tipo de servicio) y 2.4.6 (procedimiento).

Un tipo virtual es un tipo que tiene **virtual** o **redefined** como <virtuality>. Un tipo redefinido es un tipo que tiene **redefined** o **finalized** como <virtuality>.

Cada tipo virtual tiene asociada una limitación de virtualidad que es un <identifier> de la misma clase de entidad que el tipo virtual. Si se especifica <virtuality constraint>, la limitación de virtualidad es el <identifier> contenido, y en caso contrario la limitación de virtualidad se deriva como se describe a continuación.

Un tipo virtual y sus limitaciones no pueden tener parámetros de contexto formal.

Solamente los tipos virtuales pueden tener <virtuality constraint> especificada.

Si <virtuality> está presente en la referencia y en la definición referenciada, deben entonces ser iguales. Si <procedure preamble> está presente en ambas referencia de procedimiento y en la definición referenciada, deben ser iguales.

Un tipo virtual debe tener en sus puertas exactamente los mismos parámetros formales, puertas y señales que su limitación.

*Semántica:*

Un tipo virtual puede ser redefinido ...

El acceso a un tipo virtual mediante ...

Un tipo virtual o redefinido que no tiene <specialization> especificada, puede tener una <specialization> implícita. La limitación de virtualidad y la posible <specialization> implícita se deriva del siguiente modo:

Dado un tipo virtual V con una <virtuality constraint> VA y una especialización VS, y dado un tipo redefinido R de V con una <virtuality constraint> RA y una <specialization> RS, entonces se tiene lo siguiente (todas las reglas se aplican en el orden dado):

- si se omite VA, entonces VA es la misma que V;
- si se omite VS y VA no denota V, entonces VS es la misma que VA;
- si VS está presente, entonces VS debe ser la misma o un subtipo de VA;
- si se omite RA, entonces RA es la misma que R;
- si se omite RS, entonces RS es la misma que VA;
- RA debe ser la misma o un subtipo de VA;
- RS debe ser la misma o un subtipo de RA;
- si R es un tipo virtual, se aplican las mismas reglas para R que para V.

Un subtipo de un tipo virtual es un subtipo del tipo virtual original y no de una redefinición posible.

## 15 Mantenimiento de SDL

Esta cláusula describe la terminología y las reglas para el mantenimiento de SDL acordadas en la reunión de la Comisión de Estudio 10 de noviembre de 1993, y el "procedimiento de petición de cambio" asociado.

## 15.1 Terminología

**15.1.1** **Error** es una inconsistencia interna dentro de la Recomendación Z.100.

**15.1.2** Una **corrección textual** es un cambio en el texto o en los diagramas de la Recomendación Z.100 que corrige errores de copia o tipográficos.

**15.1.3** Un **ítem abierto** es un tema identificado pero no resuelto. Un ítem abierto puede identificarse, bien por una petición de cambio bien por acuerdo de la Comisión de Estudio o Grupo de Trabajo.

**15.1.4** Una **deficiencia** es un tema identificado en el que las semánticas de SDL no están (claramente) definidas por la Recomendación Z.100.

**15.1.5** Una **clarificación** es un cambio en el texto o en los diagramas de la Recomendación Z.100 que clarifica texto o diagramas anteriores que pueden tener una comprensión ambigua sin esta clarificación. La clarificación debe tratar de hacer que la Recomendación Z.100 corresponda a la semántica de SDL tal como es entendido por la Comisión de Estudio o Grupo de Trabajo.

**15.1.6** Una **modificación** es un cambio en el texto o en los diagramas de la Recomendación Z.100 que cambia la semántica de SDL.

**15.1.7** Una **característica desacordada** es una característica de SDL que debe ser eliminada de SDL en la próxima revisión de la Recomendación Z.100.

**15.1.8** Una **ampliación** es una característica nueva que no debe cambiar la semántica de características definida en la Recomendación Z.100.

## 15.2 Reglas de mantenimiento

En el texto siguiente deben considerarse las referencias a la Recomendación Z.100 que incluyen este addendum y la Recomendación Z.105.

1) Cuando se detecta un error o deficiencia en la Recomendación Z.100, debe ser corregido o clarificado. La corrección de un error debe implicar los menores cambios posibles. Las correcciones de errores y las clarificaciones se introducirán en la lista básica de cambios para la Recomendación Z.100 y entrarán en vigor de inmediato.

2) Salvo para correcciones de errores y resolución de ítems abiertos desde el periodo de estudio anterior, las modificaciones y ampliaciones a SDL solamente pueden ser tomadas en consideración como resultado de una petición de cambio formulada por un grupo de usuarios importante. Una petición de cambio debe ir seguida de una investigación por la Comisión de Estudio o Grupo de Trabajo en colaboración con representantes del grupo de usuarios peticionarios, de modo que la necesidad y el beneficio de tal modificación queden claramente establecidos y se tenga la certeza de que una característica existente de SDL es inadecuada.

3) Las modificaciones y ampliaciones que no se deriven de correcciones de errores, serán divulgadas ampliamente, y los puntos de vista de los usuarios y de los fabricantes de herramientas serán debidamente escrutados antes de adoptar el cambio. A menos que existan circunstancias especiales que requieran que tales cambios se implanten lo más pronto posible, estos no serán recomendados hasta que lo haga la Recomendación Z.100.

4) Hasta la publicación de una Recomendación Z.100 revisada se mantendrá una lista básica de cambios de la Recomendación Z.100 que abarque a la Recomendación Z.100 y todos los anexos, excepto la definición formal. Los apéndices, addenda o suplementos serán publicados según lo decida la Comisión de Estudio. Para asegurar una distribución efectiva de la lista básica de cambios de la Recomendación Z.100 ésta se publicará como informes COM y por medios electrónicos apropiados.

5) Para las deficiencias en la Recomendación Z.100 debe consultarse la definición formal. Esto puede conducir a una clarificación o una corrección, lo que se registra en la lista básica de cambios de la Recomendación Z.100.

## 15.3 Procedimiento de petición de cambios

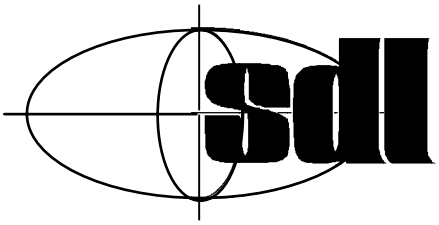
El procedimiento de petición de cambio tiene por objeto que los usuarios de SDL dentro y fuera del UIT-T puedan formular preguntas sobre el significado preciso de la Recomendación Z.100, hagan sugerencias sobre cambios en SDL o la Recomendación Z.100 y aporten a su vez información de retorno sobre cambios propuestos a SDL. Los cambios propuestos a SDL serán publicados por el Grupo de Expertos sobre SDL antes de ser implantados.

Las peticiones de cambio deben utilizar el formulario de petición de cambio o proporcionar la información listada en el mismo. La clase de petición debe ser indicada claramente (corrección de error, clarificación, simplificación, ampliación, modificación o característica desacordada). Es también importante que para cualquier cambio distinto a una corrección de error se indique el número de usuarios que apoya la petición.

Todas las peticiones de cambio deben ser tratadas en reuniones de la Comisión de Estudio del UIT-T responsable para la Recomendación Z.100. En el caso de correcciones y clarificaciones, los cambios pueden introducirse en la lista de correcciones sin consultar a los usuarios. En los demás casos se compila una lista de ítems abiertos. La información debe ser distribuida a los usuarios:

- como informes contribuciones blancas del UIT-T;
- a través del Newsletter SDL (ISSN 1023-7151);
- por correo electrónico a las listas de correo electrónico de SDL ( tal como "SDL_News");
- por otros medios acordados por los expertos de la Comisión de Estudio 10.

Las reacciones de los usuarios serán evaluadas por expertos de la Comisión de Estudio 10 para determinar el grado de apoyo u oposición a cada cambio. Un cambio solamente puede incorporarse a la lista de cambios aceptada si hay un apoyo importante por parte de los usuarios y solo existen objeciones serias a la propuesta por parte de unos pocos. Finalmente, todos los cambios aceptados será incorporados a la Recomendación Z.100 revisada. Los usuarios deben ser conscientes de que hasta que los cambios hayan sido incorporados y aprobados por la Comisión de Estudio responsable para la Recomendación Z.100, no son recomendados por el UIT-T.



## Formulario de petición de cambio

| | | |
|---|---|---|
| Por favor, rellene los siguientes detalles | | |
| Carácter del cambio: | q corrección de error<br>q simplificación<br>q modificación | q clarificación<br>q ampliación<br>q retirada de comisión |
| Breve resumen de la petición de cambio | | |
| Breve justificación de la petición de cambio | | |
| ¿Es este punto de vista compartido en su organización? | q sí | q no |
| ¿Ha consultado Vd. a otros usuarios? | q sí | q no |
| ¿A cuántos usuarios Vd. representa? | q 1-5<br>q 11-100 | q 6-10<br>q sobre 100 |
| Su nombre y dirección | | |

*por favor, adjunte más hojas en caso necesario*

SDL (Z.100) Rapporteur, c/o ITU-T, Place des Nations, CH-1211, Geneva 20, Switzerland.
Fax: +41 22 730 5853, e-mail: SDL.rapporteur@itu.ch

## SERIES DE RECOMENDACIONES DEL UIT-T

| | |
|---|---|
| Serie A | Organización del trabajo del UIT-T |
| Serie B | Medios de expresión: definiciones, símbolos, clasificación |
| Serie C | Estadísticas generales de telecomunicaciones |
| Serie D | Principios generales de tarificación |
| Serie E | Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos |
| Serie F | Servicios de telecomunicación no telefónicos |
| Serie G | Sistemas y medios de transmisión, sistemas y redes digitales |
| Serie H | Sistemas audiovisuales y multimedios |
| Serie I | Red digital de servicios integrados |
| Serie J | Transmisiones de señales radiofónicas, de televisión y de otras senales multimedios |
| Serie K | Protección contra las interferencias |
| Serie L | Construcción, instalación y protección de los cables y otros elementos de planta exterior |
| Serie M | Mantenimiento: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales |
| Serie N | Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión |
| Serie O | Especificaciones de los aparatos de medida |
| Serie P | Calidad de transmisión telefónica, instalaciones telefónicas y redes locales |
| Serie Q | Conmutación y señalización |
| Serie R | Transmisión telegráfica |
| Serie S | Equipos terminales para servicios de telegrafía |
| Serie T | Terminales para servicios de telemática |
| Serie U | Conmutación telegráfica |
| Serie V | Comunicación de datos por la red telefónica |
| Serie X | Redes de datos y comunicación entre sistemas abiertos |
| **Serie Z** | **Lenguajes de programación** |