UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T Z.120

(10/96)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE Z: LENGUAJES DE PROGRAMACIÓN

Técnicas de descripción formal – Gráficos de secuencias de mensajes

---

Gráficos de secuencias de mensajes

Recomendación UIT-T Z.120
Reemplazada por una versión más reciente

(Anteriormente Recomendación del CCITT)

Reemplazada por una versión más reciente

RECOMENDACIONES DE LA SERIE Z DEL UIT-T

LENGUAJES DE PROGRAMACIÓN

Lenguaje de especificación y descripción (LED)	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.

Reemplazada por una versión más reciente

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.120 ha sido revisada por la Comisión de Estudio 10 (1993-1996) del UIT-T y fue aprobada por la CMNT (Ginebra, 9-18 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

Página

ALCANCE/OBJETIVO

El objetivo de recomendar un gráfico de secuencias de mensajes es proporcionar un lenguaje de dibujo para especificar y describir el comportamiento de comunicación de componentes de sistema y su entorno mediante el intercambio de mensajes. Dado que en los MSC el comportamiento de comunicación se presenta de una manera muy intuitiva y transparente, especialmente en la representación gráfica, el lenguaje MSC se aprende, utiliza e interpreta con facilidad. Con respecto a otros lenguajes puede utilizarse para sustentar metodologías de especificación, diseño, simulación, prueba y documentación de sistemas.

COBERTURA

En esta Recomendación se presenta una definición de sintaxis para los gráficos de secuencias de mensajes en su representación abstracta, textual y gráfica. También se ofrece una descripción semántica informal.

APLICACIÓN

El campo principal de aplicación de los MSC es una especificación general del comportamiento de comunicación de los sistemas en tiempo real, en particular, los sistemas de conmutación de telecomunicaciones. Mediante los dibujos de sistema seleccionados de los MSC se pueden especificar primordialmente las situaciones "normalizadas". Los casos no normalizados que comprenden comportamientos excepcionales se pueden describir a partir de los normalizados. De este modo, pueden utilizarse los MSC para la especificación de requisitos, la especificación de interfaces, la simulación y validación, la especificación de casos de prueba y la documentación para sistemas en tiempo real. Pueden utilizarse los MSC junto con otros lenguajes de especificación, en particular el SDL. En este contexto, los MSC proporcionan también una base para el diseño de sistemas en SDL.

SITUACIÓN/ESTABILIDAD

La situación de los MSC básicos es estable, incluidos los elementos constructivos para los casos, la creación y terminación de casos, el intercambio de mensajes, las acciones, el tratamiento de temporizadores y las condiciones. Se prevén ulteriores desarrollos para nuevos conceptos estructurales como las corregiones generalizadas, expresión en línea, referencia a los MSC y HMSC.

DOCUMENTACIÓN CONEXA

– Recomendación UIT-T Q.65 (1997), *Metodología funcional y unificada para la caracterización de servicios y capacidades de red.*

– Recomendación UIT-T X.210 (1993), *Tecnología de la información – Interconexión de sistemas abiertos – Modelo de referencia básico: Convenios para la definición de servicios de interconexión de sistemas abiertos. (Texto común UIT-T | ISO/CEI.)*

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

Reemplazada por una versión más reciente

Recomendación Z.120

GRÁFICOS DE SECUENCIAS DE MENSAJES

(revisada en 1996)

1 Introducción

1.1 Introducción al MSC

Un gráfico de secuencias de mensajes (MSC, *message sequence chart*) muestra las secuencias de mensajes intercambiados entre componentes de sistema y su entorno. En SDL (Recomendación Z.100, UIT 1996), los componentes de sistema son servicios, procesos y bloques. Los MSC se han venido utilizando desde hace tiempo por las Comisiones de Estudio del UIT-T (anteriormente CCITT) en sus Recomendaciones así como en la industria, de acuerdo con diferentes convenios y bajo diversos nombres, tales como gráfico secuencial de señales, diagrama de flujos de información, flujos de mensajes y diagrama de flechas.

El motivo de normalizar los MSC es proporcionarles el apoyo de medios, intercambiar los MSC entre medios diferentes, facilitar la correspondencia con las especificaciones SDL y armonizar su utilización dentro de la UIT.

Una parte del trabajo de normalización consiste en proporcionar una definición clara del significado de un MSC, lo que se hace en la presente Recomendación [anexo B (1995)] proporcionando una semántica formal basada en el álgebra de procesos.

Otra forma de expresar el significado de los MSC puede ser relacionarlos con especificaciones SDL, como sigue: Un MSC describe una o más trazas de una especificación de sistema SDL. En consecuencia, puede obtenerse un MSC de una especificación de sistema en SDL existente y luego se puede utilizar como ejemplo para anotar el resultado de una animación.

Debido a la normalización, ha aumentado considerablemente la importancia de los MSC para la ingeniería de los sistemas. Por consiguiente los MSC pueden servir como:

a) una visión de conjunto de un servicio ofrecido por varias entidades;

b) declaración para la especificación de requisitos;

c) base para la formulación de especificaciones de SDL;

d) base para la simulación y la validación de sistemas;

e) base para la selección y especificación de casos de prueba;

f) especificación de comunicación;

g) especificación de interfaz;

h) formalización de la utilización de casos en el análisis y diseño orientado a objetos.

Como, generalmente, un MSC únicamente describe un comportamiento parcial, la selección de comportamientos parciales reviste suma importancia. Los candidatos a MSC son, principalmente, situaciones "normalizadas", a partir de las cuales se construyen por lo general otras situaciones y se tratan comportamientos excepcionales, por ejemplo, los ocasionados por errores de diversos tipos.

A continuación, se presenta una sintaxis para los gráficos de secuencias de mensajes en representación abstracta, textual y gráfica. Se proporciona asimismo la descripción semántica (verbal) informal correspondiente.

Esta Recomendación se estructura de la siguiente manera: En la cláusula 2, se exponen las reglas generales de sintaxis, dibujo y paginación. La cláusula 3 contiene la definición de sintaxis del documento de gráficos de secuencias de mensajes, que es una colección de gráficos de secuencias de mensajes. La cláusula 4 contiene la definición sintáctica de los gráficos de secuencias de mensajes y las reglas sintácticas de los constituyentes básicos, es decir caso, mensaje, entorno, ordenación general, condición, temporizador, acción, creación y terminación de casos. En la cláusula 5, se presentan conceptos de alto nivel relacionados con la estructuración y la modularización. Estos conceptos sustentan una especificación en orden descendente y permiten el perfeccionamiento de los casos individuales mediante una corregión (véase 5.1) y la descomposición de casos (véase 5.2). Los conceptos más avanzados – expresión en línea (véase 5.3), referencia de MSC (véase 5.4), MSC de alto nivel (véase 5.5) – permiten la composición de MSC y la reutilización de (parte de) MSC con diferentes niveles. En la cláusula 6, se ofrecen ejemplos de todos los elementos constructivos de MSC. El anexo A contiene un índice de las <palabras clave> y no terminales.

1.2 Meta-lenguaje para la gramática textual

En la forma Backus-Naur (BNF, *Backus-Naur form*) se indica un símbolo terminal no encerrándolo en paréntesis angulares (que son los signos menor que, y mayor que, < y >) o mediante una de las dos representaciones <nombre> y <cadena de caracteres>. Obsérvese que los dos terminales especiales <nombre> y <cadena de caracteres> pueden también tener semánticas acentuadas como se define más adelante.

Los paréntesis angulares y las palabras interiores a los mismos pueden ser o símbolos no terminales o uno de los terminales <cadena de caracteres> o <nombre>. Las categorías sintácticas son no terminales indicados por una o más palabras situadas dentro de paréntesis angulares. Para cada símbolo no terminal, se facilita una regla de producción en forma de gramática textual concreta o en forma de gramática gráfica. Por ejemplo:

<msc inst interface> ::=
inst <instance list> <end>

Una regla de producción para un símbolo no terminal se compone del símbolo no terminal situado a la izquierda del signo ::=, y de uno o más constructivos constituidos por símbolos no terminales y/o terminales colocados a la derecha de ese signo. Por ejemplo, <msc inst interface> e <instance list> del ejemplo anterior son no terminales; **inst** es un símbolo terminal.

A veces el símbolo incluye una parte subrayada. Esta parte subrayada acentúa un aspecto semántico de ese símbolo, por ejemplo <msc name> es sintácticamente idéntico a <name> pero semánticamente requiere que el nombre sea un nombre de gráfico de secuencias de mensajes.

A la derecha del signo ::= puede haber varias producciones alternativas del no terminal, separadas mediante barras verticales (|). Por ejemplo:

<incomplete message area> ::=
<lost message area>
| <found message area>

expresa que un <incomplete message area> es una <lost message area> o una <found message area>.

Pueden agruparse conjuntamente elementos sintácticos utilizando llaves ({ y }). Un grupo encerrado entre llaves puede contener una o más barras verticales, lo que indica la presencia de elementos sintácticos alternativos. Por ejemplo:

<msc document body> ::=
{ <message sequence chart> | <msc diagram> }*

La repetición de grupos encerrados entre llaves se representa mediante un asterisco (*) o un signo más (+). Un asterisco indica que el grupo es facultativo y puede repetirse ulteriormente cualquier número de veces. Un signo más expresa que el grupo puede estar presente y puede repetirse ulteriormente cualquier número de veces. El ejemplo anterior indica que un <msc document body> puede estar vacío pero puede también contener cualquier número de <message sequence chart> y/o <msc diagram>.

Si los elementos sintácticos se agrupan mediante corchetes ([y]) entonces ese grupo es opcional. Por ejemplo:

<identifier> ::=
[<qualifier>] <name>

expresa que un <identifier> puede contener, aunque no necesariamente <qualifier>.

1.3 Meta-lenguaje para la gramática gráfica

El meta-lenguaje para la gramática gráfica se basa en algunos constructivos que se describen informalmente a continuación. La sintaxis gráfica no es lo suficientemente precisa para describir gráficos en los que no hay variaciones gráficas. Se admiten pequeñas variaciones de las formas reales de los símbolos terminales gráficos, que incluyen por ejemplo, el oscurecimiento de símbolos rellenados, la forma de una cabecera de flecha y el tamaño relativo de los elementos gráficos. Cuando sea necesario, se completará la sintaxis gráfica con la explicación informal de las apariencias de las construcciones.

El meta-lenguaje consiste en una notación del tipo BNF con las metaconstrucciones especiales: ***contains***, ***is followed by***, ***is associated with***, ***is attached to***, ***above*** y ***set***. Estos constructivos se comportan como reglas de producción BNF normales pero implican, adicionalmente, alguna relación geométrica o lógica entre los argumentos. El constructivo ***is attached to*** se comporta de modo ligeramente diferente como se explica a continuación. El primer miembro de todos los constructivos, salvo ***above***, debe ser un símbolo. Un símbolo es un no terminal que produce en cada secuencia de producción un terminal gráfico exactamente. Consideraremos un símbolo ***anexo a (attached to)*** a otras zonas o que ***está asociado a (is associated with)*** una cadena de texto también como un símbolo. La explicación es informal y el meta-lenguaje no describe con precisión las dependencias geométricas.

contains:

"<area1> ***contains*** <area2>" significa que <area2> está contenido, en principio, geométricamente, pero también lógicamente dentro de <area1>. Véase la figura 1-1.

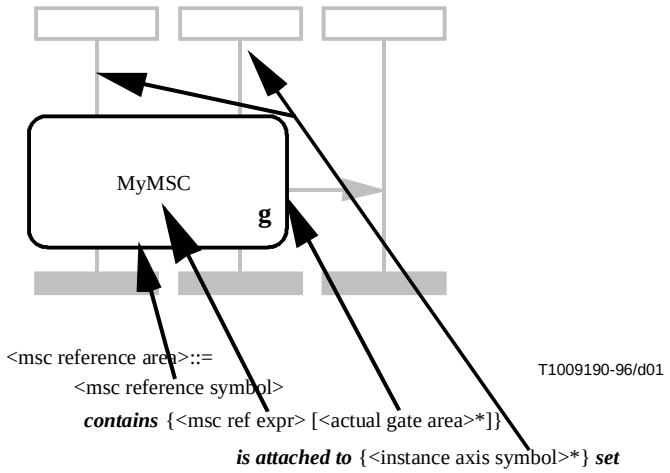


Figura 1-1/Z.120 – Ejemplo correspondiente a "contains"

is followed by:

"<area1> ***is followed by*** <area2>" significa que <area2> está relacionado geométrica y lógicamente con <area1>. Véase la figura 1-2.

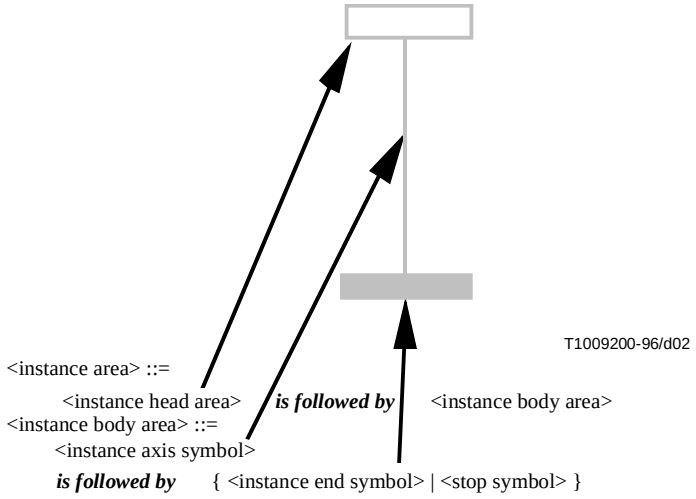


Figura 1-2/Z.120 – Ejemplo correspondiente a "is followed by"

is associated with:

"<area1> ***is associated with*** <area2>" significa que <area2> se extenderá a una cadena de texto afiliada con <area1>. No hay ninguna asociación geométrica más estrecha entre las zonas que la que indica que deben considerarse asociadas entre sí. Véase la figura 1-3.

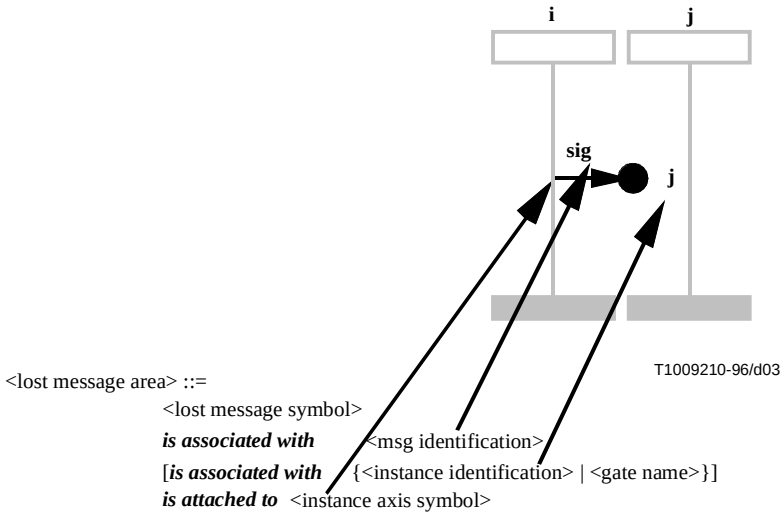
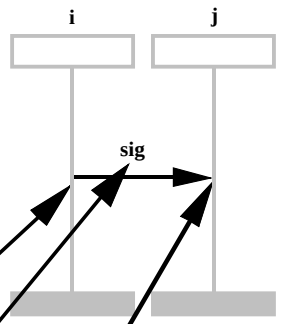


Figura 1-3/Z.120 – Ejemplo correspondiente a "is associated with"

is attached to:

"<area1> ***is attached to*** <area2>" no es como una regla de producción BNF normal estando limitada su utilización. <area2> debe ser un símbolo o un conjunto de símbolos. El significado de la regla de producción "<P>::=<area1> ***is attached to*** <area2>" es que si un no terminal <P> se desarrolla utilizando esta regla, únicamente se produce el símbolo <area1>. En vez de producirse también <area2>, se identifica una aparición de <area2> debida a una de las demás producciones. El resultado del constructivo ***is attached to*** es una relación lógica y geométrica entre <area1> y el símbolo <area2> para la aparición implicada. En el caso en que el primer miembro sea un conjunto de símbolos, hay una relación entre el símbolo <area1> y todos los elementos del conjunto. Véase la figura 1-4.



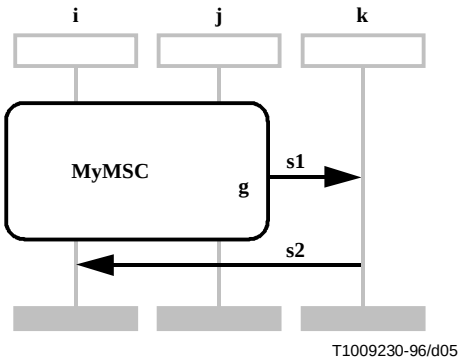
<message area> ::=
<message symbol>
is associated with <msg identification>
is attached to { <message start area> <message end area> }

-- *Obsérvese que si el símbolo A* ***está agregado al*** *símbolo B, el símbolo B también* ***está agregado al*** *símbolo A.*
En consecuencia, en la gramática se define siempre dos veces la agregación entre dos símbolos.
-- *Seguiremos uno de los dos no terminales anteriores para ver esto:*
<message end area> ::=
<message in area> | <gate in area> | <gate out def area> | <inline expr>
<message in area> ::=
<message in symbol>
is attached to <instance axis symbol>
is attached to <message symbol>
<message in symbol> ::=
<void symbol>
En consecuencia un <message symbol> ***is attached to*** *un* <message in symbol> y vice versa.

Figura 1-4/Z.120 – Ejemplo correspondiente a "is attached to"

above:

"<area1> ***above*** <area2>" significa que <area1> y <area2> están ordenados lógicamente. Esto se expresa de forma geométrica ordenando verticalmente <area1> y <area2>. Si dos <area> tienen la misma coordenada vertical no se define ninguna ordenación entre ellas, salvo que la salida de un mensaje debe producirse antes de su entrada. Véase la figura 1-5.



<event layer> ::=

<event area> | <event area> ***above*** <event layer>

La figura describe una sucesión de eventos. Cuando los eventos corresponden a casos distintos, la coordenada vertical geométrica carece de significado semántico.
La figura anterior describe la siguiente secuencia:
i,j: **reference** mr1: MyMSC **gate** g **output** s1 **to** k;
k: **input** s1 **from** mr1 **via** g;
k: **output** s2 **to** i;
i: **input** s2 **from** k.

Figura 1-5/Z.120 – Ejemplo correspondiente a "above"

set:

El metasímbolo ***set*** es un operador posterior que actúa sobre los elementos sintácticos inmediatamente precedentes encerrados entre llaves e indica un conjunto (desordenado) de elementos. Cada elemento puede ser cualquier grupo de elementos sintácticos, en cuyo caso debe desarrollarse antes de la aplicación del metasímbolo ***set***.

Ejemplos

<text layer> ::=

{<text area>*} ***set***

es un conjunto de cero e más <text area>.

<msc body area> ::=

{ <instance layer> <text layer> <gate def layer> <event layer> <connector layer> } ***set***

es un conjunto desordenado de elementos situados dentro de las llaves.

2 Reglas generales

2.1 Reglas de léxico

Las reglas de léxico definen unidades de léxico. Las unidades de léxico son los símbolos terminales de la *Sintaxis textual concreta*.

<lexical unit> ::= <word>
| <character string>
| <special>
| <composite special>
| <note>
| <keyword>

```
<word> ::=              {<alphanumeric> | <full stop>}*
                        <alphanumeric>
                        {<alphanumeric> | <full stop>}*

<alphanumeric> ::=      <letter>
                   |    <decimal digit>
                   |    <national>

<letter> ::=            A   | B   | C   | D   | E   | F   | G   | H   | I   | J   | K   | L   | M
                   |    N   | O   | P   | Q   | R   | S   | T   | U   | V   | W   | X   | Y   | Z
                   |    a   | b   | c   | d   | e   | f   | g   | h   | i   | j   | k   | l   | m
                   |    n   | o   | p   | q   | r   | s   | t   | u   | v   | w   | x   | y   | z

<decimal digit> ::=     0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9

<national> ::=          #   | ‘   | ¤   | @   | \
                   |    <left square bracket>
                   |    <right square bracket>
                   |    <left curly bracket>
                   |    <vertical line>
                   |    <right curly bracket>
                   |    <overline>
                   |    <upward arrow head>

<left square bracket> ::=   [

<right square bracket> ::=  ]

<left curly bracket> ::=    {

<vertical line> ::=         |

<right curly bracket> ::=   }

<overline> ::=              ~

<upward arrow head> ::=     ^

<full stop> ::=             .

<underline> ::=             _

<character string> ::=  <apostrophe>
                        {<alphanumeric>
                   |    <other character>
                   |    <special>
                   |    <full stop>
                   |    <underline>
                   |    <space>
                   |    <apostrophe><apostrophe>}*
                   |    <apostrophe>

<text> ::=              { <alphanumeric>
                   |    <other character>
                   |    <special>
                   |    <full stop>
                   |    <underline>
                   |    <space>
                   |    <apostrophe> }*

<apostrophe> ::=    '

<other character> ::= ?     | &   | %   | +   | -   | !   | /   | >   | *   | "   | <   | =

<special> ::=           (   | )   | ,   | ;   | :
```

<composite special> ::= << | >>

<note> ::= /* <text> */

<name> ::= <word> { <underline> <word> }*

<keyword> ::= **action**
| **all**
| **alt**
| **as**
| **before**
| **begin**
| **block**
| **by**
| **comment**
| **concurrent**
| **condition**
| **connect**
| **create**
| **decomposed**
| **empty**
| **end**
| **endconcurrent**
| **endinstance**
| **endmsc**
| **endexpr**
| **env**
| **exc**
| **expr**
| **external**
| **found**
| **from**
| **gate**
| **in**
| **inf**
| **inline**
| **inst**
| **instance**
| **loop**
| **lost**
| **msc**
| **mscdocument**
| **msg**
| **opt**
| **order**
| **out**
| **par**
| **process**
| **reference**
| **related**
| **reset**
| **service**
| **seq**
| **set**
| **shared**
| **stop**
| **subst**
| **system**

| **text**
| **timeout**
| **to**
| **via**

Los caracteres <national> se representan en la lista anterior según la versión de referencia del alfabeto internacional número 5 del CCITT (Recomendación T.50). La responsabilidad de la definición de representaciones nacionales de esos caracteres compete a las entidades de normalización nacionales.

Los caracteres de control se definen como en la Recomendación T.50. Puede existir una secuencia de caracteres de control donde aparezca un <space>, con el mismo significado que un <space>. El <space> representa el carácter espacio del alfabeto número 5 del CCITT.

La aparición de un carácter de control no es significativa para <note>. Un carácter de control no puede aparecer en cadenas literales de caracteres si su presencia es significativa.

Todas las <lexical unit> salvo <character string>, <letter> se tratan siempre como si fueran mayúsculas. (El tratamiento de <national> puede definirse por las entidades de normalización nacionales.)

Una <lexical unit> se concluye mediante el primer carácter que no puede ser parte de <lexical unit> según la sintaxis especificada anteriormente. Cuando un carácter <underline> va seguido de uno o más <space>, se pasan por alto todos esos caracteres (incluido el <underline>), esto es A_B designa el mismo <name> que AB. Esta utilización de <underline> permite la división de varios <lexical unit> sobre más de una línea.

Cuando el carácter / va seguido inmediatamente de un * fuera de una <note>, inicia una <note>. El carácter * seguido inmediatamente del carácter / en una <note> siempre termina la <note>. Una <note> puede insertarse antes o después de cualquier <lexical unit>.

2.2 Reglas de visibilidad y denominación

Las entidades se identifican y se referencian mediante nombres asociados. Las entidades se agrupan en clases de entidad para dar flexibilidad a las reglas de denominación. Existen las siguientes clases de entidad:

a) documento MSC;

b) MSC;

c) caso;

d) condición;

e) temporizador;

f) mensaje;

g) puerta.

La unidad de ámbito para un MSC es el documento MSC. Dos entidades dentro de una unidad de ámbito y que pertenecen a la misma clase de entidad no pueden tener el mismo nombre. Las ocurrencias diferentes de un nombre de condición, nombre de temporizador y nombre de mensaje dentro de una unidad de ámbito denotan la misma unidad. El nombre de una entidad es visible dentro de la unidad de ámbito pero no fuera de ella. Solo se pueden utilizar nombres visibles al referirse a entidades.

Los nombres de puerta van siempre asociados con un MSC específico. En consecuencia, los mismos nombres de puerta pueden aplicarse a MSC diferentes.

2.3 Comentario

Hay tres clases de comentarios.

En primer lugar está *note* que aparece únicamente en la sintaxis textual (véanse las reglas de léxico).

En segundo lugar está *comment* que es una notación para representar explicaciones informales asociadas con símbolos o con texto.

La *sintaxis textual concreta* de los comentarios es:

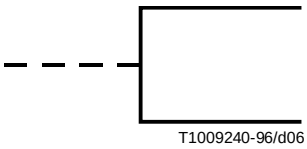
<end> ::= [<comment>]**;**

<comment> ::= **comment** <character string>

En la *gramática gráfica concreta* se utiliza la sintaxis siguiente:

<comment area> ::= <comment symbol> ***contains*** <text>

<comment symbol> ::=



Un <comment symbol> puede agregarse a los siguientes símbolos gráficos: <text symbol>, <instance head symbol>, <instance end symbol>, <message out symbol>, <message in symbol>, <lost message symbol>, <found message symbol>, <general order symbol>, <condition symbol>, <set symbol>, <reset symbol>, <timeout symbol>, <action symbol>, <createline symbol>, <stop symbol>, <coregion symbol>, <inline expression symbol>, <separator symbol>, <msc reference symbol>, <hmsc start symbol>, <hmsc end symbol>, <par frame symbol>, <connection point symbol>.

En tercer lugar está *text* que puede utilizarse para comentarios globales.

El texto en la *gramática textual* se define como sigue:

<text definition> ::= **text** <character string> <end>

El texto en la *gramática gráfica* se define como sigue:

<text area> ::= <text symbol> ***contains*** <text>

<text symbol> ::=

2.4 Reglas de dibujo

El usuario puede elegir el tamaño de los símbolos gráficos. Las fronteras de los símbolos no deben superponerse ni cruzarse. Son excepciones a esta regla:

a) el cruce de símbolo de mensaje y símbolo de ordenación general con símbolo de mensaje, símbolo de ordenación general, líneas en los símbolos de temporizador, símbolo de creación de línea, símbolo de eje de caso, línea de trazos de un símbolo de comentario y símbolo de condición;

b) el cruce líneas de símbolos de temporización con símbolo de mensaje, símbolo de ordenación general, líneas de símbolos de temporizador, símbolo de creación de línea, línea de trazos de un símbolo de comentario y símbolo de condición;

c) intersección entre un símbolo creación de línea con un símbolo mensaje, símbolo de ordenación general, líneas de símbolos de temporizador, símbolo de eje de caso y línea de trazos de un símbolo de comentario;

d) intersección de símbolo de condición con símbolo de eje de caso, símbolo de mensaje, símbolo de ordenación general, líneas de símbolos de temporización;

e) intersección de símbolo de expresión en línea con símbolo de eje de caso;

f) intersección de un símbolo de referencia con un símbolo de eje de caso;

g) intersección de un símbolo de acción con un símbolo de eje de caso en forma de línea y la superposición de un símbolo de acción con un símbolo de eje de caso en forma de columna;

h) intersección de símbolo de línea hmsc con símbolo de línea hmsc.

Hay dos formas del símbolo de eje de caso y del símbolo de corregión: forma de renglón y forma de columna. No se permite mezclar ambas formas dentro de un caso, salvo un eje de línea única con una columna para formar corregiones.

Si una condición compartida (véase 4.6) atraviesa un caso no implicado en esa condición, el eje de caso se dibuja a través de ésta.

Si una referencia compartida (véase 5.4) atraviesa un caso no implicado en esa referencia, el eje de caso se dibuja a través de ésta.

Si una expresión en línea compartida (véase 5.3) atraviesa un caso no implicado en esa expresión en línea, no puede dibujarse a través del eje de caso.

Cuando el símbolo de eje de caso tiene forma de columna, los bordes verticales del símbolo de acción tienen que coincidir con las líneas de la columna.

Las líneas de mensaje pueden ser horizontales o diagonales descendentes (con respecto al sentido de la flecha) y pueden estar formadas por una secuencia de segmentos rectos conectados.

Si en el mismo punto de un eje de caso hay un suceso de entrada y un suceso de salida se considera que el suceso de entrada está dibujado por encima del suceso de salida. A este punto no se le puede añadir una ordenación general. No se permite el dibujo de dos o más sucesos de salida en el mismo punto. No se permite dibujar dos o más sucesos de entrada en el mismo punto. Los sucesos que siguen son sucesos de entrada: entrada de mensaje, mensaje encontrado y temporización. Los siguientes sucesos son sucesos de salida: salida de mensaje, mensaje perdido, ajuste de temporizador, reiniciación de temporizador y creación de caso.

2.5 Paginación de los MSC

Los MSC pueden dividirse entre varias páginas. La subdivisión puede ser horizontal y vertical. Alternativamente, puede evitarse la partición mediante la composición de MSC o la descomposición de casos (véase la cláusula 5).

Cuando el MSC se divide verticalmente en varias páginas, se repite el <msc heading> en cada página, pero el símbolo de fin de caso únicamente puede aparecer en una página (en la "última" página del caso en cuestión). Para cada caso, debe aparecer <instance head area> en la primera página donde comienza el caso y deberá repetirse con guiones en cada una de las páginas siguientes en las que continua.

Si los mensajes, temporizadores, sentencias creación o condiciones continúan de una página a la siguiente, el texto completo asociado al mensaje, temporizador, creación o condición deberá estar presente en la primera página y repetirse la totalidad o parte del texto en la siguiente página.

En las páginas de un MSC debe incluirse la numeración de páginas para indicar la posición correcta de las páginas. Las páginas deben numerarse con dos símbolos: "v-h", donde "v" es el número de la página vertical y "h" el número de la página horizontal. Se utilizarán caracteres numéricos arábigos para los números verticales y letras mayúsculas del alfabeto inglés ("A" a "Z") para los horizontales. Si no es suficiente la gama "A"-"Z" se ampliará con las parejas "AA" a "AZ", "BA" a "BZ", etc.

3 Documento de gráficos de secuencias de mensajes

La cabecera del documento de gráficos de secuencias de mensajes contiene el nombre del documento y, facultativamente, a continuación de la palabra clave **related to**, el identificador (nombre de trayecto) del documento SDL al que se refiere el MSC.

Gramática concreta

<msc document> ::=

<msc document head> <msc document body>

<msc document head> ::=

<document head> | <document head area>

<msc document body> ::=

{ <message sequence chart> | <msc diagram> }*

Gramática textual concreta

<document head> ::=

mscdocument <msc document name> [**related to** <sdl reference>] <end>

<sdl reference> ::= <sdl document identifier>

<identifier> ::= [<qualifier>] <name>

<qualifier> ::= **<<** <text> **>>**

El texto de un identificador no debe contener "<<" o ">>".

Gramática gráfica concreta

<document head area> ::=

<frame symbol> ***contains*** <document head>

Semántica

Un documento de gráficos de secuencias de mensajes es una colección de gráficos de secuencias de mensajes que se refieren facultativamente a un documento SDL correspondiente.

4 MSC básico

4.1 Gráfico de secuencias de mensajes

Un gráfico de secuencias de mensajes, abreviado normalmente mediante MSC, describe un flujo de mensajes entre casos. Un gráfico de secuencias de mensajes describe un comportamiento parcial de un sistema. Si bien es obvio que el nombre de *gráfico de secuencias de mensajes* se debe a su representación gráfica, se utiliza tanto para la representación gráfica como para la textual.

La cabecera del gráfico de secuencias de mensajes está constituida por el nombre del MSC y (facultativamente) por una lista de los casos contenidos en el MSC y una lista de las puertas del MSC.

Un MSC se describe mediante casos y sucesos o mediante una expresión que relaciona referencias de MSC sin citar explícitamente los casos [véase 5.5 MSC de alto nivel (HMSC, *high-level MSC*)].

Gramática textual concreta

<message sequence chart> ::=

msc <msc head> { <msc body> | **expr** <msc expression> } **endmsc** <end>

<msc head> ::= <msc name> <end> [<msc interface>]

<msc interface> ::= [<msc inst interface>] [<msc gate interface>]

<msc inst interface> ::=

inst <instance list> <end>

<instance list> ::= <instance name> [**:** <instance kind>] [**,** <instance list>]

<instance kind> ::= [<kind denominator>] <identifier>

<kind denominator> ::=
system | **block** | **process** | **service** | <name>

<msc gate interface> ::=
<msc gate def>*

<msc gate def> ::= **gate** { <msg gate> | <order gate> } <end>

<msg gate> ::= <def in gate> | <def out gate>

<order gate> ::= <def order in gate> | <def order out gate>

<msc body> ::= <msc statement>*

<msc statement> ::=
<text definition> | <event definition> | <old instance head statement> <instance event list>

<event definition> ::=
<instance name> **:** <instance event list> | <instance name list> **:** <multi instance event list>

<instance event list> ::=
{ <instance event> <end> }+

<instance event> ::=
<orderable event> | <non-orderable event>

<orderable event> ::=
[<event name>]
{ <message event> | <incomplete message event> | <create> | <timer statement> | <action> }
[**before** <event name list>]

Cuando el suceso está generalmente ordenado se utiliza el opcional <event name>.

<event name list> ::=
{ <event name> | <gate name> } [, <event name list>]

El <gate name> se refiere a una <def order out gate>, <actual order in gate>, <inline order out gate> o <def order out gate>. El <event name> se refiere a un <orderable event>.

<non-orderable event> ::=
<coregion> | <shared condition> | <shared msc reference> | <shared inline expr>
| <instance head statement> | <instance end statement> | <stop>

<instance name list> ::=
<instance name> { **,** <instance name> }* | **all**

<multi instance event list> ::=
{ <multi instance event> <end> }+

<multi instance event> ::=
<condition> | <msc reference> | <inline expr>

<old instance head statement> ::=
instance <instance name> [[**:**] <instance kind>] [<decomposition>] <end>

El no terminal <old instance head statement> únicamente está previsto para la compatibilidad regresiva con las descripciones MSC-92 existentes.

Para cada <instance head statement> u <old instance head statement> debe existir también un suceso <instance end statement> o <stop> correspondiente. Para cada caso no deben existir sucesos antes de que se defina su <instance head statement>. Para cada caso no debe haber sucesos después de su <instance end statement>. Para cada caso no debe haber más de un <instance head statement> ni tampoco más de un <instance end statement>. Para cada caso no debe existir ningún <instance head statement> detrás de <old instance head statement>.

El <instance list> de la <msc interface>, si está presente, debe tener los mismos casos que los especificados en el <msc body>.

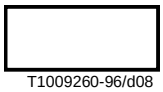
Gramática gráfica concreta

<msc diagram> ::= <msc symbol> ***contains*** { <msc heading> { <msc body area> | <mscexpr area> } }

<msc symbol> ::= <frame symbol>

is attached to { <def gate area>* } ***set***

<frame symbol> ::=



T1009260-96/d08

<msc heading> ::= **msc** <msc name>

<msc body area> ::=

{ <instance layer> <text layer> <gate def layer> <event layer> <connector layer> } ***set***

<instance layer> ::= { <instance area>* } ***set***

<text layer> ::= { <text area>* } ***set***

<gate def layer> ::= { <def gate area>* } ***set***

<event layer> ::= <event area> | <event area> ***above*** <event layer>

<connector layer> ::=

{ <message area>* | <incomplete message area>* } ***set***

<event area> ::= <instance event area>

| <shared event area>

| <create area>

<instance event area> ::=

<message event area>

| <timer area>

| <concurrent area>

| <action area>

<shared event area> ::=

| <condition area>

| <msc reference area>

| <inline expression area>

Semántica

Un MSC describe la comunicación entre cierto número de componentes de sistema y entre estos componentes y el resto del mundo, denominado entorno. Para cada componente de sistema abarcado por un MSC hay un eje de caso. La comunicación entre componentes de sistema se efectúa mediante mensajes. El envío y el consumo de mensajes son dos eventos asíncronos. Se supone que el entorno de un MSC es capaz de recibir y enviar mensajes desde y hacia el gráfico de secuencias de mensajes; no se supone ninguna ordenación de los eventos de mensaje dentro del entorno. Aunque el comportamiento del entorno es no determinístico, se supone que obedece a las limitaciones impuestas por el gráfico de secuencias de mensajes.

No se supone ningún eje temporal global para un MSC. El tiempo transcurre de arriba hacia abajo a lo largo de cada eje de caso, pero no se supone ninguna escala temporal propia. Si no se introduce la corregión o la expresión en línea (véanse 5.1, 5.3), se supone una ordenación temporal total de eventos a lo largo de cada eje de caso. Los eventos de casos distintos se ordenan mediante mensajes – un mensaje debe enviarse antes de su consunción – o mediante el denominado mecanismo de ordenación generalizada. Utilizando este mecanismo de ordenación generalizada, pueden ordenarse de forma explícita los "eventos ordenables" de casos distintos (incluso en MSC diferentes). No se prescribe ningún otro tipo de ordenación. Por consiguiente, un gráfico de secuencias de mensajes impone una ordenación parcial al conjunto de eventos contenidos. Se denomina ordenación parcial a una relación binaria que es transitiva, antisimétrica y reflexiva.

Para las entradas de mensaje [rotuladas por in(mi)] y las salidas de mensaje [rotuladas por out(mi)] del MSC de la figura 4-1 a) se obtiene la relación de ordenación siguiente: out(m2) < in(m2), out(m3) < in(m3), out(m4) < in(m4), in(m1) < out(m2) < out(m3) < in(m4), in(m2) < out(m4) junto con el cierre transitivo.

La ordenación parcial se puede describir en forma mínima (sin una representación explícita del cierre transitivo), mediante su grafo de conectividad [figura 4-1 b)].

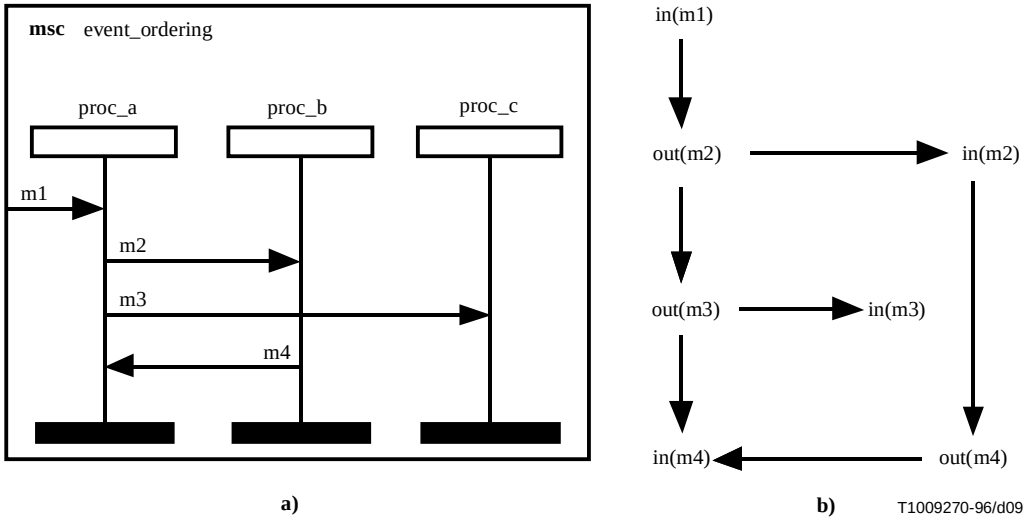


Figura 4-1/Z.120 – Gráfico de secuencias de mensajes y grafo de continuidad correspondiente

En el anexo B (1995) se facilita la semántica formal de los MSC basada en el álgebra de procesos. La semántica de un MSC se puede relacionar con la semántica del SDL mediante la noción de grafo de alcanzabilidad. Cada secuenciación de un MSC describe un trazado de un nodo a otro nodo (o conjunto de nodos) del gráfico de alcanzabilidad que describe el comportamiento de una especificación de sistema SDL. El grafo de alcanzabilidad consta de nodos y bordes. Los nodos designan estados globales del sistema. Un estado global del sistema está determinado por los valores de las variables, el estado de ejecución de cada proceso y el contenido de las colas de mensajes. Los bordes corresponden a los eventos ejecutados por el sistema, por ejemplo el envío y consumo de un mensaje o la ejecución de una tarea. Una secuenciación de un MSC designa una ordenación total de eventos compatible con la ordenación parcial definida por el MSC.

En la representación textual, la <event definition> la proporciona un <instance name> seguido de la <event list> anexa o una <instance name list> seguida del <multi instance event> anexo, empleándose como separador el signo :. El no terminal <instance event> designa o un evento anexo a un caso aislado, por ejemplo <action> o un objeto compartido, <shared condition> en el que se utiliza la palabra clave **shared** junto con la lista de casos o la palabra clave **all** para designar el conjunto de casos entre los cuales se comparte la condición. Un objeto compartido puede representarse alternativamente mediante <multi instance event>. Se han previsto las distintas notaciones para facilitar, por una parte una descripción *orientada a casos* y por otra una descripción *orientada a eventos*. Pueden combinarse, arbitrariamente, ambas notaciones. La descripción *orientada a casos* enumera eventos asociados a un caso. Dentro de la representación textual *orientada a eventos*, pueden enumerarse los eventos en forma de una traza de ejecución posible y no ordenarse respecto a los casos.

El opcional <msc interface> que describe la interfaz del MSC con su entorno consta del <msc inst interface> y del <msc gate interface>. El <msc inst interface> proporciona una declaración de un caso, por ejemplo <instance name> y, facultativamente, <instance kind>. Como normalmente un MSC consta únicamente de una sección de ejecución de un sistema el <msc inst interface> describe los puntos de conexión de casos con el entorno. El <msc gate interface> proporciona una definición de un mensaje y las puertas de ordenación contenidas en el MSC. Las puertas de mensaje definen los puntos de conexión de los mensajes con el entorno. Facultativamente pueden asociarse con las puertas, nombres de puertas.

4.2 Caso

Un gráfico de secuencias de mensajes se compone de casos de entidades interactuantes. Un caso de una entidad es un objeto que tiene las propiedades de esa entidad. Con relación al SDL, una entidad puede ser un proceso, bloque o servicio SDL. En la cabecera de un caso, además del nombre del caso puede especificarse el nombre de la entidad, por ejemplo el nombre del proceso. En el cuerpo del caso se especifica la ordenación de los eventos.

Gramática textual concreta

<instance head statement> ::=

instance [<instance kind>] [<decomposition>]

<instance end statement> ::=

endinstance

Gramática gráfica concreta

<instance area> ::= <instance head area> ***is followed by*** <instance body area>

<instance head area> ::=

<instance head symbol>

is associated with <instance heading>

[***is attached to*** <createline symbol>]

<instance heading> ::=

<instance name> [[**:**] <instance kind>] [<decomposition>]

<instance head symbol> ::=

<instance body area> ::=

<instance axis symbol>

is followed by { <instance end symbol> | <stop symbol> }

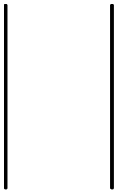
<instance axis symbol> ::=

{ <instance axis symbol1> | <instance axis symbol2> }

is attached to { <event area> * } ***set***

<instance axis symbol1> ::=

<instance axis symbol2> ::=



<instance end symbol> ::=

T1009280-96/d10

La <instance heading> puede colocarse encima o dentro del <instance head symbol> o puede dividirse de modo que el <instance name> esté colocado dentro del <instance head symbol> y la <instance kind> y <decomposition> estén por encima. En este último caso, el símbolo : es facultativo (y deberá estar encima, si está presente).

Semántica

Los casos se definen dentro del cuerpo del gráfico de secuencias de mensajes. El símbolo de fin de caso determina el fin de la descripción del caso dentro del MSC. No describe la terminación del caso (véase 4.10 parada de caso). Consiguientemente, el símbolo de cabecera del caso determina el comienzo de la descripción de caso dentro del MSC. No describe la creación del caso (véase 4.9 creación de caso).

En el contexto del SDL, un caso puede referirse a un proceso (palabra clave **process**), a un servicio (palabra clave **service**) o a un bloque (palabra clave **block**). Fuera del SDL puede referirse a cualquier clase de entidad. La definición de caso proporciona una descripción de evento para entradas y salidas de mensajes, acciones, condiciones compartidas y locales, temporizador, creación y parada de casos. Fuera de las corregiones (véase 5.1) y de las expresiones en línea (véase 5.3) se supone una ordenación total de los eventos a lo largo de cada eje de caso. Dentro de las corregiones no se supone ninguna ordenación temporal de eventos si no se han prescrito constructivos de sincronización ulteriores en forma de relaciones de orden general.

4.3 Mensaje

Un mensaje dentro de un MSC es una relación entre una salida y una entrada. La salida puede provenir del entorno (a través de una puerta) o de un caso y la entrada se dirige hacia el entorno (una puerta) o un caso.

Si se pierden la salida o la entrada, se produce un mensaje incompleto que está representado únicamente por un solo evento.

Un mensaje intercambiado entre dos casos se puede dividir en dos eventos: la entrada de mensaje y la salida de mensaje; por ejemplo, el segundo mensaje de la figura 4-1 a) puede dividirse en out(m2) (salida) e in(m2) (entrada). En la representación textual, la entrada de mensaje está representada por la palabra clave **in**, y la salida de mensaje por la palabra clave **out** ambas seguidas por el nombre del mensaje y, facultativamente, un nombre de caso de mensaje. A un mensaje pueden asignársele parámetros entre paréntesis.

La correspondencia entre salidas de mensaje y entradas de mensaje debe definirse de manera unívoca. En la representación textual, normalmente la correspondencia entre entradas y salidas se desprende de la identificación de nombre de mensaje y de la identificación de dirección. En la representación gráfica, un mensaje se representa por una flecha.

La pérdida de un mensaje, es decir el caso en que se envía un mensaje pero no se consume, puede indicarse por la palabra clave **lost** en la representación textual y mediante un cuadrado negro en la representación gráfica.

De modo simétrico, un mensaje encontrado espontáneamente, es decir un mensaje que aparece sin provenir de ninguna parte puede definirse mediante la palabra clave **found** en la representación textual y mediante un cuadrado blanco en la representación gráfica.

Puede definirse una ordenación temporal de eventos de mensajes de casos distintos mediante la palabra clave **before** en la representación textual. En la representación gráfica, los constructivos de sincronización definen estos conceptos de ordenación generalizada en forma de líneas de conexión.

Gramática textual concreta

<message event> ::=
<message output> | <message input>

<message output> ::=
out <msg identification> **to** <input address>

<message input> ::=
in <msg identification> **from** <output address>

<incomplete message event> ::=
<incomplete message output> | <incomplete message input>

<incomplete message output> ::=
out <msg identification> **to lost** [<input address>]

<incomplete message input> ::=
in <msg identification> **from found** [<output address>]

<msg identification> ::=
<message name> [**,** <message instance name>] [(<parameter list>)]

<parameter list> ::=
<parameter name> [**,** <parameter list>]

<output address> ::=
<instance name> | { **env** | <reference identification>} [**via** <gate name>]

<reference identification> ::=
reference <msc reference identification> | **inline** <inline expr identification>

El <gate name> se refiere a una <def in gate>. Si únicamente se utiliza la palabra clave **env** ello implica que <output address> designa una <def in gate> que tiene un nombre implícito (dado por la <msg identification> correspondiente y la dirección **in**).

<input address> ::=
<instance name> | { **env** | <reference identification>} [**via** <gate name>]

El <gate name> se refiere a una <def out gate>. Si únicamente se utiliza la palabra clave **env** ello implica que <input address> designa una <def out gate> que tiene un nombre implícito (dado por la <msg identification> correspondiente y la dirección **out**).

Para los mensajes intercambiados entre casos se deben respetar las siguientes reglas: para cada <message output> hay que especificar un <message input> correspondiente y viceversa. Cuando <message name> y <address> no basten para establecer una correspondencia unívoca hay que utilizar <message instance name>.

No se permite que <message output> dependa causalmente de su <message input> a través de otros mensajes o constructivos de ordenación general. Esto sucede cuando el grafo de conectividad (véase 4.1) contiene bucles. Si se especifica una <parameter list> para un <message input> habrá que especificarla también para el <message output> correspondiente y viceversa. Las <parameter list> tienen que ser idénticas.

Gramática gráfica concreta

<message event area> ::=

{ <message out area> | <message in area> }

{***is followed by*** <general order area> }*

{***is attached to*** <general order area> }*

Los eventos de mensajes pueden ordenarse en general, según distintas relaciones de ordenación general. Los eventos de mensaje aparecen a cada lado de la relación de orden.

<message out area> ::=

<message out symbol>

is attached to <instance axis symbol>

is attached to <message symbol>

<message out symbol> ::=

<void symbol>

<void symbol> ::= .

El <void symbol> es un punto geométrico sin extensión espacial.

NOTA 1 – El <message out symbol> es realmente un punto situado en el eje de caso. El final de un símbolo de mensaje que carezca de cabeza de flecha está también situado en este punto del eje de caso.

<message in area> ::=

<message in symbol>

is attached to <instance axis symbol>

is attached to <message symbol>

<message in symbol> ::=

<void symbol>

NOTA 2 – El <message in symbol> es realmente un punto situado en el eje de caso. El final de un símbolo de mensaje que carezca de cabeza de flecha está también situado en este punto del eje de caso.

<message area> ::= <message symbol> ***is associated with*** <msg identification>

is attached to { <message start area> <message end area> }

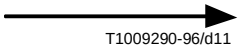
<message start area> ::=

<message out area> | <actual out gate area> | <def in gate area> | <inline gate area>

<message end area> ::=

<message in area> | <actual in gate area> | <def out gate area> | <inline gate area>

<message symbol> ::=



T1009290-96/d11

NOTA 3 – Se autoriza la imagen especular de <message symbol>. El punto de la cabecera de flecha debe estar situado en el eje de caso.

NOTA 4 – En la representación gráfica, no es necesario el nombre de caso de mensaje para una descripción sintáctica unívoca.

<incomplete message area> ::=

{ <lost message area> | <found message area> }

{ ***is followed by*** <general order area> }*

{ ***is attached to*** <general order area> }*

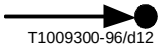
<lost message area> ::=

<lost message symbol> ***is associated with*** <msg identification>

[***is associated with*** { <instance name> | <gate name> }]

is attached to <message start area>

<lost message symbol> ::=



NOTA 5 – El <lost message symbol> describe el evento del lado de salida, es decir la línea continua comienza en la <message start area> donde se produce el evento. El blanco facultativo previsto del mensaje puede indicarse mediante un identificador asociado al símbolo. La identificación del blanco debe escribirse cerca del círculo negro, mientras que la identificación del mensaje debe escribirse cerca de la flecha.

NOTA 6 – Puede también utilizarse la imagen especular del símbolo.

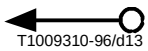
<found message area> ::=

<found message symbol> ***is associated with*** <msg identification>

[***is associated with*** { <instance name> | <gate name> }]

is attached to <message end area>

<found message symbol> ::=



NOTA 7 – El <found message symbol> describe el evento del lado de entrada (cabecera de flecha) que debería estar en una <message end area>. El caso o puerta que, presuntamente, era el origen del mensaje se indica mediante la identificación facultativa proporcionada por el texto asociado al círculo del símbolo. La identificación del mensaje deberá escribirse cerca de la parte de la flecha.

NOTA 8 – Puede también utilizarse la imagen especular del símbolo.

Semántica

En un MSC, la salida de mensaje indica el envío del mensaje (correspondiente a una salida SDL), la entrada indica el consumo del mensaje (correspondiente a una entrada SDL). No se proporciona ningún constructivo para la recepción del mensaje (entrada en la memoria tampón). Tampoco se agregan definiciones de tipo a los parámetros de la lista de parámetros.

Un mensaje incompleto es un mensaje salida (cuya entrada se ha perdido) o un mensaje entrada (cuya salida es desconocida).

En el caso en que los eventos de mensaje coincidan con otros eventos, véanse las reglas de dibujo de 2.4.

4.4 Entorno y puertas

Las puertas representan la interfaz entre el MSC y su entorno. Cualquier mensaje o relación de orden unida a la trama del MSC constituye una puerta.

Toda puerta de mensaje siempre tiene un nombre. El nombre puede definirse explícitamente mediante un nombre asociado a la puerta en el marco. En otros casos puede darse el nombre de forma implícita mediante la dirección del mensaje a través de la puerta y el nombre del mensaje, por ejemplo "in_X" para una puerta que recibe un mensaje X de su entorno.

Se utilizan puertas de mensaje cuando se sitúan referencias al MSC en un contexto más amplio de otro MSC. Las puertas reales de la referencia al MSC se conectan entonces a las otras puertas de mensajes o casos. De forma similar a las definiciones de puertas, las puertas reales pueden tener nombres implícitos o explícitos.

Las puertas de orden representan relaciones de orden no completadas en las que un evento interior al MSC se ordenará con relación a un evento del entorno. Las puertas de orden siempre se denominan de forma explícita. Se considera que las puertas de orden tienen un sentido desde el evento ordenado en primer lugar al evento que le sigue.

Las puertas de orden se usan también en referencias a MSC en otros MSC. Las puertas de orden reales de la referencia del MSC se conectan a otras puertas de orden o a eventos.

Las puertas de las expresiones en línea son similares a las puertas de los marcos de MSC y referencias de MSC residiendo la diferencia en el hecho de que el marco de la expresión en línea sirve como marco de la definición de puerta (del interior) y como símbolo de utilización de la puerta real (en el exterior).

Las puertas no conectadas a una referencia MSC se definen implícitamente para propagar el siguiente marco de cierre (ya sea un marco MSC o un marco de expresión en línea). Una puerta propagada deberá tener el mismo nombre de puerta que la puerta en forma propagada. Esto es una abreviatura práctica que ahorra una línea de conexión que de otro modo contribuiría a emborronar el diagrama.

Las puertas de las expresiones en línea son simplemente puntos de tránsito en el marco de la expresión en línea. Si la puerta no continúa fuera del marco, se aplican las siguientes reglas implícitas.

1) Si hay otras puertas con el mismo nombre de la misma expresión en línea, la continuación indicada para una de las puertas se mantiene para todas las demás.

2) Si no hay otras puertas con el mismo nombre y no existe continuación, se supone una continuación implícita con el siguiente marco de cierre (marco de MSC o marco de expresión en línea). Véase la figura 4-2.

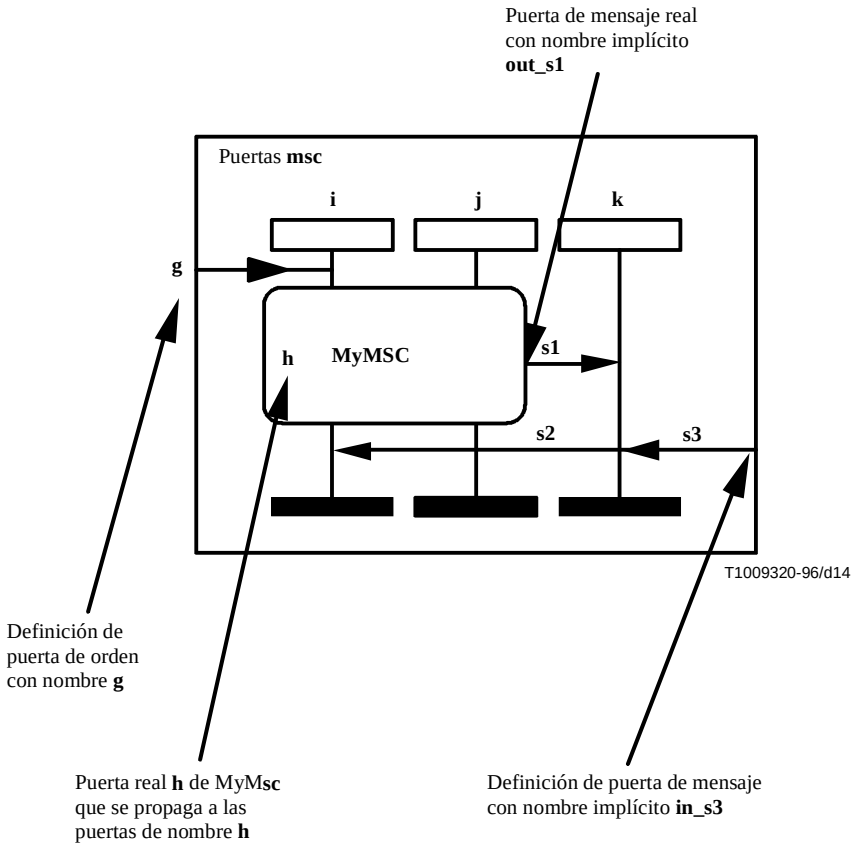


Figura 4-2/Z.120 – Ejemplo de puertas

Gramática textual concreta

<actual out gate> ::=
[<gate name>] **out** <msg identification> **to** <input dest>

<actual in gate> ::= [<gate name>] **in** <msg identification> **from** <output dest>

<input dest> ::= **lost** [<input address>] | <input address>

<output dest> ::= **found** [<output address>] | <output address>

<def in gate> ::= [<gate name>] **out** <msg identification> **to** <input dest>

<def out gate> ::= [<gate name>] **in** <msg identification> **from** <output dest>

Si se omite <gate name> se supone un nombre implícito dado por la dirección y la <msg identification> correspondiente.

<actual order out gate> ::=
<gate name> **before** <order dest>

<order dest> ::=
<event name> | { **env** | <reference identification>} **via** <gate name>

El <event name> se refiere a un evento ordenable. El <gate name> se refiere a <def order out gate>, <actual order in gate>, o <inline order in gate>.

<actual order in gate> ::=
<gate name>

<def order in gate> ::=
<gate name> **before** <order dest>

El primer <gate name> define el nombre de esta puerta de orden.

<def order out gate> ::=
<gate name>

El <gate name> define el nombre de esta puerta de orden.

<inline out gate> ::=
<def out gate>
[**external out** <msg identification> **to** <input dest>]

<inline in gate> ::= <def in gate>
[**external in** <msg identification> **from** <output dest>]

<inline order out gate> ::=
<gate name>
[**external before** <order dest>]

<inline order in gate> ::=
<gate name> **before** <order dest>
[**external**]

Gramática gráfica concreta

<inline gate area> ::=
{ <inline out gate area> | <inline in gate area> }
[***is associated with*** <gate identification>]

<inline out gate area> ::=

<void symbol>

is attached to <inline expression symbol>

[***is attached to*** { <message symbol> | <found message symbol> }]

[***is attached to*** { <message symbol> | <lost message symbol> }]

<inline in gate area> ::=

<void symbol>

is attached to <inline expression symbol>

[***is attached to*** { <message symbol> | <lost message symbol> }]

[***is attached to*** { <message symbol> | <found message symbol> }]

Una puerta de expresión en línea está normalmente unida a un símbolo de mensaje situado dentro del marco de expresión en línea y a un símbolo de mensaje exterior al marco de expresión en línea. Cuando no hay ningún símbolo en el interior de la puerta, ello significa que se trata de un mensaje incompleto asociado a la puerta. Cuando no hay ningún mensaje fuera de la puerta, ello significa que la puerta prolifera al siguiente marco.

<inline order gate area> ::=

<inline order out gate area> | <inline order in gate area>

<inline order out gate area> ::=

<void symbol>

is attached to <inline expression symbol>

is attached to <general order symbol>

[***is attached to*** <general order symbol>]

El primer <general order symbol> es una relación de ordenación general interior a la expresión en línea de forma tal que el evento interior a la expresión es anterior a la puerta. El <general order symbol> facultativo, se refiere a una relación de orden general unida a la puerta exterior a la expresión en línea.

<inline order in gate area> ::=

<void symbol>

is attached to <inline expression symbol>

is attached to <general order symbol>

[***is attached to*** <general order symbol>]

El primer <general order symbol> es una relación de ordenación general interior a la expresión en línea tal que la puerta es anterior al evento interior a la expresión. El <general order symbol> facultativo, se refiere a una relación de orden general unida a la puerta exterior a la expresión en línea.

<def gate area> ::= { <def out gate area> | <def in gate area> | <def order out gate area> | <def order in gate area> }

is attached to <msc symbol>

<def out gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to { <message symbol> | <found message symbol> }

NOTA 1 – El <message symbol> o el <found message symbol> deben tener el extremo de la cabecera de la flecha unido al <def out gate area>.

<def in gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to { <message symbol> | <lost message symbol> }

NOTA 2 – El <message symbol> o el <lost message symbol> deben tener su extremo abierto unido al <def in gate area>.

<def order out gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to <general order area>

<def order in gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is followed by <general order area>

<gate identification> ::=

<gate name>

<actual gate area> ::=

<actual out gate area> | <actual in gate area> | <actual order out gate area>

| <actual order in gate area>

<actual out gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to <msc reference symbol>

[***is attached to*** { <message symbol> | <lost message symbol> }]

NOTA 3 – El <actual out gate area> está unido al extremo abierto del <message symbol> o el <lost message symbol>.

<actual in gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to <msc reference symbol>

[***is attached to*** { <message symbol> | <found message symbol> }]

NOTA 4 – El <actual in gate area> está unido a la punta de la flecha del <message symbol> o el <found message symbol>.

<actual order out gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to <msc reference symbol>

is followed by <general order area>

<actual order in gate area> ::=

<void symbol> [***is associated with*** <gate identification>]

is attached to <msc reference symbol>

is attached to <general order area>

Semántica estática

Se admite que los <gate name> de un MSC sean ambiguos, aunque las referencias a <gate name> ambiguas son ilegales.

4.5 Ordenación general

Se utiliza la ordenación general para indicar que dos eventos están ordenados en el tiempo, aunque esto no puede deducirse de la ordenación impuesta por los mensajes.

Gramática textual completa

La gramática textual se indica en 4.1.

Gramática gráfica concreta

<general order area> ::=

<general order symbol>

is attached to <ordered event area>

<general order symbol> ::=

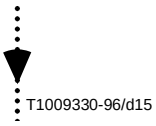
<general order symbol1> | <general order symbol2>

<general order symbol1> ::=

NOTA 1 – Se admite que el <general order symbol1> tenga una forma escalonada. Ello implica que puede estar constituido por segmentos verticales y horizontales consecutivos. Los segmentos de <general order symbol1> pueden superponerse, pero ello únicamente se admite si pueden determinarse de forma inequívoca los <general order symbol1> afectados. Esto significa que no pueden estar implicados nuevos órdenes.

El <general order symbol1> únicamente puede tener lugar dentro de un <instance axis symbol2> (forma columna). Las líneas de conexión desde el <general order symbol1> a las <ordered event area> que ordenan, deben ser horizontales.

<general order symbol2> ::=



NOTA 2 – El <general order symbol2> puede tener cualquier orientación e incluso ser una línea quebrada.

<ordered event area> ::=

<actual order in gate area>
| <actual order out gate area>
| <def order in gate area>
| <def order out gate area>
| <inline order gate area>
| <message event area>
| <incomplete message area>
| <timer area>
| <create area>
| <action symbol>

Semántica estática

El orden parcial de los eventos definidos por los constructivos de ordenación general y los mensajes debe ser irreflexivo.

Semántica

Los símbolos de orden general describen órdenes entre eventos que de otra forma no estarían ordenados. Esto es particularmente útil cuando los eventos de una corregión deben estar más ordenados que todas las permutaciones posibles.

Ejemplo de ordenación general

En la figura 4-3 se representan cuatro ejemplos que describen la misma ordenación general. Se presentan los siguientes órdenes: a anterior a b, a anterior a d, c anterior a b, c anterior a d, a y c están desordenados y b y d están desordenados.

4.6 Condición

Una condición describe o bien un estado de sistema global (condición global) que se refiere a todos los casos contenidos en el MSC o bien a un estado que se refiere a un subconjunto de casos (condición no global). En el segundo caso, la condición puede ser local, es decir, estar unida a un solo caso. En la representación textual la condición tiene que definirse para cada caso al que está unida utilizando la palabra clave **condition** junto con el nombre de la condición. Si la condición se refiere a varios casos, entonces la palabra clave **shared** junto con la lista de casos designa el conjunto de casos que comparten la condición. Puede definirse una condición global que se refiere a todos los casos mediante la palabra clave **shared all**.

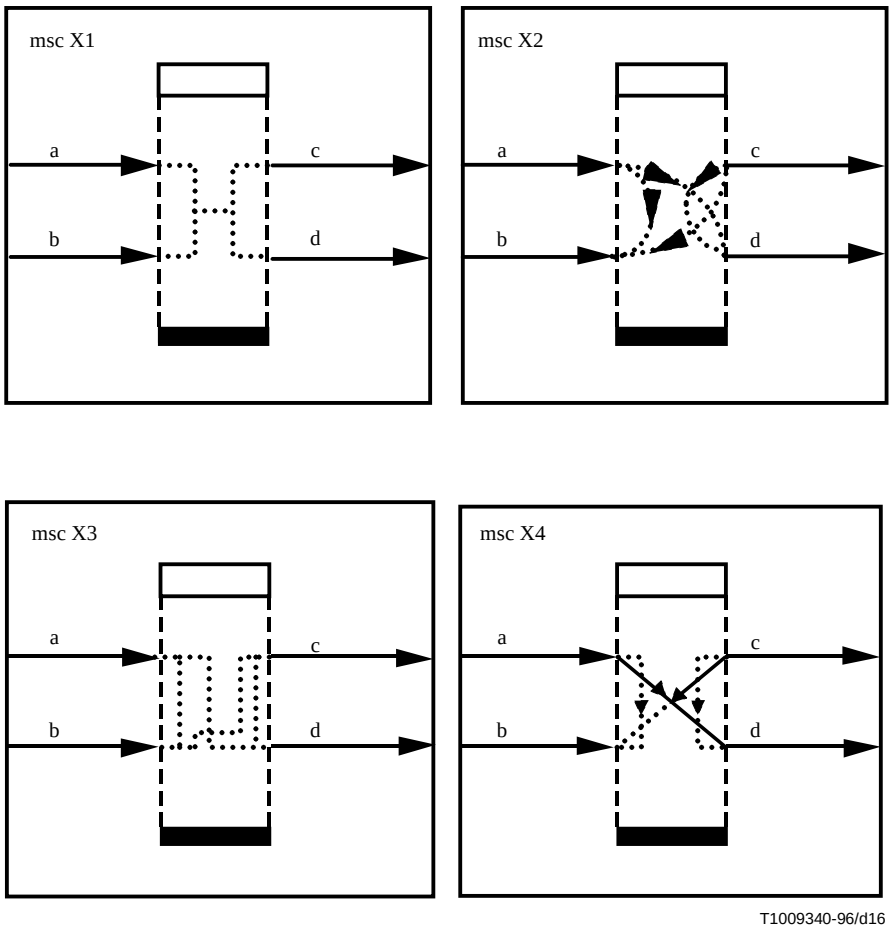


Figura 4-3/Z.120 – Ejemplo de ordenación general

Pueden utilizarse condiciones globales para limitar la composición de MSC en MSC de alto nivel. Esto se examina ulteriormente en 5.5 "MSC de alto nivel (HMSC)".

Gramática textual concreta

<shared condition> ::=

<condition identification> <shared>

<condition identification> ::=

condition <condition name list>

<condition name list> ::=

<condition name> { , <condition name> }*

<shared> ::= **shared** { [<shared instance list>] | **all** }

<shared instance list> ::=

<instance name> [**,** <shared instance list>]

<condition> ::= <condition identification>

Para cada <instance name> contenido en una <shared instance list> de una <condition>, debe especificarse un caso con la <condition> compartida correspondiente. Si el caso *b* está contenido en la <shared instance list> de una <condition> compartida unida al caso *a*, entonces el caso *a* debe estar contenido en la <shared instance list> de la <condition>

compartida correspondiente, unida al caso *b*. Si los casos *a* y *b* comparten la misma <condition>, en cada mensaje intercambiado entre estos casos, <message input> y <message output> se deben colocar antes o después de <condition>. Si dos condiciones están ordenadas directamente (porque tienen un caso en común) o indirectamente a través de condiciones sobre otros casos, este orden debe respetarse en todos los casos que comparten esas dos condiciones.

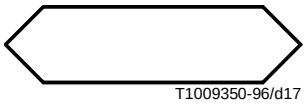
Gramática gráfica concreta

<condition area> ::=

<condition symbol> ***contains*** <condition name list>

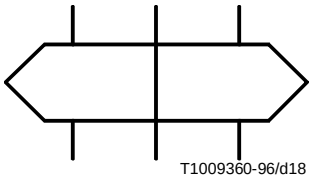
is attached to { <instance axis symbol>* } ***set***

<condition symbol> ::=



T1009350-96/d17

La <condition area> puede referirse a un solo caso o estar unida a varios casos. Si una <condition> compartida atraviesa un <instance axis symbol> no afectado por esta condición, se dibuja a través el <instance axis symbol>.



T1009360-96/d18

Semántica

Las condiciones globales que representan estados globales de sistema, se refieren a todos los casos involucrados en el MSC. Mediante la palabra clave **shared all**, en la representación textual, se puede especificar para cada MSC:

- una condición global inicial (estado inicial global);
- una condición global final (estado final global); y
- condiciones globales intermedias (estados intermedios globales).

Las condiciones globales inicial, intermedia y final no se introducen simplemente con fines de documentación en el sentido de comentarios o ilustraciones. Las condiciones globales determinan posibles continuaciones de gráficos de secuencias de mensajes que contienen el mismo conjunto de casos mediante la identificación del nombre de la condición. De este modo pueden utilizarse las condiciones para limitar posibles composiciones de MSC. La composición real de los MSC se define en forma de HMSC (véase 5.5). Las composiciones prescritas deben concordar con las condiciones contenidas en los MSC, de conformidad con la definición de continuaciones admitidas.

4.7 Temporizador

En los MSC se puede especificar la fijación de un temporizador y una temporización subsiguiente debida a la expiración del temporizador o la fijación de un temporizador y una reiniciación subsiguiente del temporizador (supervisión del tiempo). Además, pueden utilizarse separadamente los constructivos de temporizador individuales – fijación de temporizador, reiniciación/temporización – por ejemplo en el caso en que se reparta entre diferentes MSC la expiración del temporizador o la supervisión temporal. En la representación gráfica, el símbolo de fijación tiene la forma de reloj de arena conectado al eje de caso con un símbolo de línea (quebrada). La temporización se describe por una flecha de mensaje dirigida al caso unido al símbolo de reloj de arena. El símbolo de reiniciación tiene la forma de una cruz conectada con el caso mediante una línea (quebrada).

La especificación del nombre de caso del temporizador y duración del temporizador es facultativa en las representaciones textual y gráfica.

Gramática textual concreta

<timer statement> ::=
<set> | <reset> | <timeout>

<set> ::= **set** <timer name> [**,** <timer instance name>] [(<duration name>)]

<reset> ::= **reset** <timer name> [**,** <timer instance name>]

<timeout> ::= **timeout** <timer name> [**,** <timer instance name>]

En el caso en que el <timer name> no baste para una correspondencia unívoca, deberá emplearse el <timer instance name>.

Gramática gráfica concreta

<timer area> ::= { <timer set area> | <timer reset area> | <timeout area> }
{ ***is followed by*** <general order area> }*
{ ***is attached to*** <general order area> }*

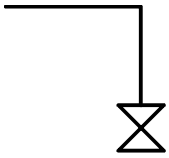
<timer set area> ::= <timer set area1> | <timer set area2>

<timer set area1> ::=
<set symbol> ***is associated with*** <timer name> [(<duration name>)]
is attached to <instance axis symbol>
[***is attached to*** {<set-reset symbol> | <reset symbol2> | <timeout symbol3> }]

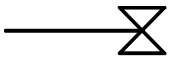
<timer set area2> ::=
<set-reset symbol> ***is associated with*** <timer name> [(<duration name>)]
is attached to <instance axis symbol>
is attached to <set symbol>
[***is attached to*** { <reset symbol2> | <timeout symbol3> }]

<set symbol> ::= <set symbol1> | < set symbol2>

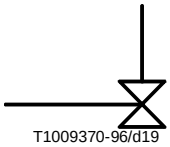
<set symbol1> ::=



<set symbol2> ::=



<set-reset symbol> ::=



<timer reset area> ::=
<timer reset area1> | <timer reset area2>

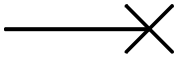
<timer reset area1> ::=
<reset symbol1> ***is associated with*** <timer name> [(<duration name>)]
is attached to <instance axis symbol>

<timer reset area2> ::=

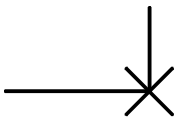
<reset symbol2> ***is associated with*** <timer name> [(<duration name>)]
is attached to <instance axis symbol>
is attached to { <set symbol> | <set-reset symbol> }

<reset symbol> ::= <reset symbol1> | <reset symbol2>

<reset symbol1> ::=



<reset symbol2> ::=



T1009380-96/d20

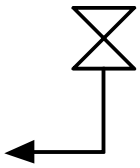
<timeout area> ::= <timeout area1> | <timeout area2>

<timeout area1>::= <timeout symbol> ***is associated with*** <timer name> [(<duration name>)]
is attached to <instance axis symbol>

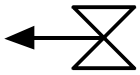
<timeout symbol> ::=

<timeout symbol1> | <timeout symbol2>

<timeout symbol1> ::=



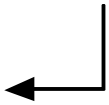
<timeout symbol2> ::=



<timeout area2> ::=

<timeout symbol3>
is attached to <instance axis symbol>
is attached to { <set symbol> | <set-reset symbol> }

<timeout symbol3> ::=



T1009390-96/d21

Semántica

La fijación y la reiniciación son constructivos de temporizador tomados del SDL. La fijación indica la puesta en marcha del temporizador y la reiniciación indica la reiniciación del temporizador. La temporización corresponde al consumo de la señal del temporizador en el SDL.

La fijación de un temporizador deberá siempre tener lugar antes de la reiniciación o temporización correspondiente.

En el caso en que los eventos de mensaje coincidan con otros eventos, véanse las reglas de dibujo de 2.4.

4.8 Acción

Además del intercambio de mensajes pueden especificarse acciones en los MSC. A las acciones se les agrega un texto informal.

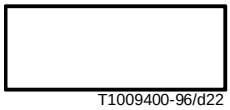
Gramática textual concreta

<action> ::= **action** <action character string>

Gramática gráfica concreta

<action area> ::= <action symbol>
is attached to <instance axis symbol>
{ ***is followed by*** <general order area> }*
{ ***is attached to*** <general order area> }*
contains <action text>

<action symbol> ::=


T1009400-96/d22

Cuando el eje de caso sea una columna, la anchura de <action symbol> debe coincidir con anchura de la columna.

Semántica

Una acción describe una actividad interna de un caso.

4.9 Creación de caso

Al igual que en el SDL, en los MSC se pueden especificar la creación y terminación de casos. Un caso puede ser creado por otro caso. No puede añadirse al caso creado ningún evento de mensaje anterior a la creación.

Gramática textual concreta

<create> ::= **create** <instance name> [(<parameter list>)]

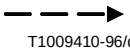
Para cada <create> debe existir un caso correspondiente con el nombre especificado. El <instance name> debe referirse a un caso con tipo proceso, si se ha especificado su tipo. Todo caso únicamente puede crearse una vez, es decir dentro de un MSC *simple*, no deben aparecer dos o más <create> con el mismo nombre.

Gramática gráfica concreta

<create area> ::= <createline symbol> [***is associated with*** <parameter list>]
is attached to { <instance axis symbol> <instance head area> } ***set***
{ ***is followed by*** <general order area> }*
{ ***is attached to*** <general order area> }*

El evento creación se designa por el final del <createline symbol> que carece de cabecera de flecha. El evento creación está unido a un eje de caso. Si la <create area> está ordenada generalmente, esta ordenación se aplica al evento creación. La cabeza de flecha apunta al <instance head symbol> del caso creado.

<createline symbol> ::=



T1009410-96/d23

Se admite la imagen especular del <createline symbol>.

Semántica

Crear define la creación dinámica de un caso por otro.

Cuando el caso creación coincida con otros eventos, véanse las reglas de dibujo de 2.4.

4.10 Parada de caso

En cierto sentido, la parada de caso es la contrapartida de la creación de caso. Sin embargo, un caso solo puede detenerse por sí mismo mientras que un caso se crea por otro caso.

Gramática textual concreta

<stop> ::= **stop**

Se admite el <stop> al final de un caso únicamente para casos de tipo proceso si se especifica el tipo.

Gramática gráfica concreta

<stop symbol> ::=

T1009420-96/d24

Semántica

La parada al final de un caso provoca la terminación de este caso.

5 Conceptos estructurales

En esta cláusula, se presentan conceptos de alto nivel que se refieren a la ordenación temporal generalizada (corregión), composición y descomposición de casos, conceptos de referencia de MSC y composición de MSC basada en el álgebra de procesos.

5.1 Corregión

En general, la ordenación total de eventos en cada caso (véase 4.1) puede no ser apropiada para las entidades que se refieren a un nivel superior al de los procesos SDL.

En consecuencia, se introduce la corregión para especificar los eventos no ordenados en un caso. La corregión trata, en particular, el caso prácticamente importante de dos o más mensajes entrantes cuya ordenación de consumo se puede intercambiar. Puede definirse una ordenación general mediante relaciones de ordenación general.

Gramática textual concreta

<coregion> ::= **concurrent** <end> <coevent>* **endconcurrent** <coevent> ::=
<orderable event> <end>

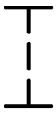
Gramática gráfica concreta

<concurrent area> ::=
<coregion symbol>
is attached to <instance axis symbol>
contains <coevent layer>

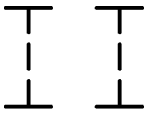
<coregion symbol> ::=
<coregion symbol1> | <coregion symbol2> <coevent layer> ::=
<coevent area> | <coevent area> ***above*** <coevent layer>

<coevent area> ::=
{ <message event area> | <incomplete message area> | <action area> |
<timer area> | <create area> }*

<coregion symbol1> ::=

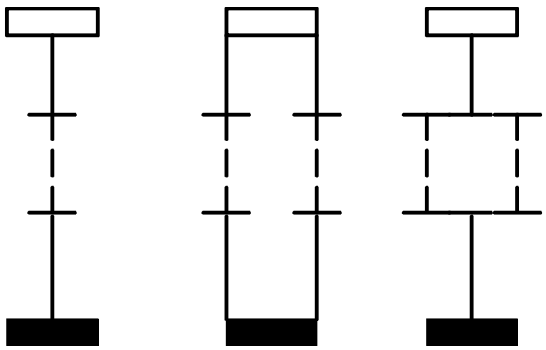


<coregion symbol2> ::=



T1009430-96/d25

Regla de dibujo: La indicación de que <coregion symbol> está unido al <instance axis symbol> significa que el <coregion symbol> debería superponerse al <instance axis symbol> como se ve en la figura 5-1.



T1009440-96/d26

Figura 5-1/Z.120 – Distintas formas de una corregión

El <coregion symbol1> no debe estar unido al <instance axis symbol2>.

Semántica

Para los MSC, se supone una ordenación temporal total de eventos dentro de cada caso. La corregión permite una excepción: los eventos contenidos en la corregión no están ordenados temporalmente si no se han previsto constructivos de sincronización ulteriores en forma de relaciones de orden general.

Si en una corregión aparece una fijación de temporizador con su temporización o reiniciación correspondientes, se mantienen las relaciones de ordenación entre estos eventos.

5.2 Descomposición de casos

En los MSC, los casos pueden referirse a entidades de distintos niveles de abstracción como se indica ya con las palabras clave (**system**, **block**, **service**, **process**). Pueden definirse operaciones de descomposición correspondientes en los casos para determinar las transiciones entre distintos niveles de abstracción. Si varios casos se componen en un solo caso, puede suavizarse la ordenación total de eventos a lo largo de este caso para mantener el comportamiento observable externamente.

Mediante la palabra clave **decomposed** y la <substructure reference> facultativa, puede añadirse a un caso un gráfico de secuencias de mensajes de perfeccionamiento.

Gramática textual concreta

<decomposition>::=
decomposed [<substructure reference>]

<substructure reference>::=
as <message sequence chart name>

Para cada caso que contenga la palabra clave **decomposed** debe especificarse el MSC de perfeccionamiento correspondiente con el nombre especificado. Para cada <message output> de un caso descompuesto, debe especificarse el <message input> correspondiente enviado al exterior del MSC de perfeccionamiento. Se mantendrá una correspondencia análoga para los mensajes entrantes.

Las expresiones en línea y referencias a MSC no deben unirse a los casos descompuestos.

Gramática gráfica concreta

La gramática gráfica figura en 4.2.

Semántica

Mediante la palabra clave **decomposed** puede añadirse a un caso un gráfico de secuencias de mensajes de perfeccionamiento. El nombre del MSC de perfeccionamiento puede definirse explícitamente mediante la <substructure reference> facultativa; en cualquier otro caso es idéntico al nombre del caso descompuesto. El MSC de perfeccionamiento representa una descomposición de este caso que no afecta a su comportamiento observable. En la representación textual, los mensajes dirigidos hacia y desde el exterior del MSC de perfeccionamiento se caracterizan por la dirección **env**, y en la representación gráfica por la conexión con el borde del MSC de perfeccionamiento (símbolo de marco). La conexión con los casos externos la proporcionan los mensajes enviados y consumidos por el caso descompuesto, utilizando la identificación de nombre de mensaje. Debe ser posible poner en correspondencia el comportamiento externo del MSC de perfeccionamiento con los mensajes del caso descompuesto. La ordenación de los eventos de mensaje especificada en un caso descompuesto debe mantenerse en el MSC de perfeccionamiento. Las acciones y condiciones dentro del MSC de perfeccionamiento pueden contemplarse con un perfeccionamiento de las acciones y condiciones del caso descompuesto. Sin embargo, y a diferencia de lo que sucede con los mensajes, no se supone una correspondencia formal con el caso descompuesto, es decir el perfeccionamiento de las acciones y condiciones no tiene por qué obedecer a reglas formales.

5.3 Expresión en línea

Mediante las expresiones del operador en línea pueden definirse dentro de un MSC la composición de estructuras de eventos. El operador se refiere a regiones alternativas de composición paralela, iteración, excepción y facultativas.

Gramática textual concreta

<shared inline expr> ::=
{ <shared loop expr> | <shared opt expr> | <shared alt expr> |
<shared par expr> | <shared exc expr> }

<shared loop expr> ::=
loop [<loop boundary>] **begin** [<inline expr identification>] <shared> <end>
[<inline gate interface>] <instance event list>
loop end

<shared opt expr> ::=
opt begin [<inline expr identification>] <shared> <end>
[<inline gate interface>] <instance event list>
opt end

<shared exc expr> ::=
exc begin [<inline expr identification>] <shared> <end>
[<inline gate interface>] <instance event list>
exc end

<shared alt expr> ::=
alt begin [<inline expr identification>] <shared> <end>
[<inline gate interface>] <instance event list>
{ **alt** <end> [<inline gate interface>] <instance event list> }*
alt end

<shared par expr> ::=
par begin [<inline expr identification>] <shared> <end>
[<inline gate interface>] <instance event list>
{ **par** <end> [<inline gate interface>] <instance event list> }*
par end

<inline expr> ::=
<loop expr> | <opt expr> | <alt expr> | <par expr> | <exc expr>

<loop expr> ::=
loop [<loop boundary>] **begin** [<inline expr identification>] <end>
[<inline gate interface>] <msc body>
loop end

<opt expr> ::=
opt begin [<inline expr identification>] <end>
[<inline gate interface>] <msc body>
opt end

<exc expr> ::=
exc begin [<inline expr identification>] <end>
[<inline gate interface>] <msc body>
exc end

<alt expr> ::=
alt begin [<inline expr identification>] <end>
<msc body>
{ **alt** <end> [<inline gate interface>] <msc body> }*
alt end

<par expr> ::=
par begin [<inline expr identification>] <end>
[<inline gate interface>] <msc body>
{ **par** <end> [<inline gate interface>] <msc body>}*
par end

<loop boundary> ::=
'<' <inf natural> [, <inf natural>] '>'

<inf natural> ::=
inf | <natural name>$^{+}$

<inline expr identification> ::=
<inline expr name>

<inline gate interface> ::=
{ **gate** <inline gate> <end>}$^{+}$

<inline gate> ::=
<inline out gate> | <inline in gate> |
<inline order out gate> | <inline order in gate>

Gramática gráfica concreta

<inline expression area> ::=
<loop area> | <opt area> | <par area> | <alt area> | <exc area>

<loop area> ::=
<inline expression symbol> ***contains***
{ **loop** [<loop boundary>] <operand area> }
is attached to { <instance axis symbol>* } ***set***
is attached to { <inline gate area>* | <inline order gate area>* } ***set***

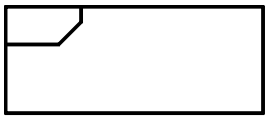
<opt area> ::=
<inline expression symbol> ***contains***
{ **opt** <operand area> }
is attached to { <instance axis symbol>* } ***set***
is attached to { <inline gate area>* | <inline order gate area>* } ***set***

<exc area> ::=
<exc inline expression symbol> ***contains***
{ **exc** <operand area> }
is attached to { <instance axis symbol>* } ***set***
is attached to { <inline gate area>* | <inline order gate area>* } ***set***

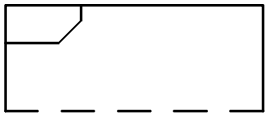
<par area> ::=
<inline expression symbol> ***contains***
{ **par** <operand area>
{ ***is followed by*** <separator area> ***is followed by*** <operand area> }* }
is attached to { <instance axis symbol>* } ***set***
is attached to { <inline gate area>* | <inline order gate area>* } ***set***

<alt area> ::=
<inline expression symbol> ***contains***
{ **alt** <operand area>
{ ***is followed by*** <separator area> ***is followed by*** <operand area> }* }
is attached to { <instance axis symbol> }* ***set***
is attached to {<inline gate area>* | <inline order gate area>* } ***set***

<inline expression symbol> ::=



<exc inline expression symbol> ::=



<operand area> ::=
{ <event layer> | <inline gate area> | <inline order gate area> }*

<separator area> ::=
<separator symbol>

<separator symbol> ::=

T1009450-96/d27

La <inline expression area> puede referirse únicamente a un caso o estar unida a varios casos. Si una expresión en línea compartida atraviesa un caso no implicado en esta expresión en línea se permite no dibujar el eje de caso a través de la expresión en línea.

Todas las expresiones de excepción deben compartirse por todos los casos del MSC.

No pueden unirse expresiones en línea a casos descompuestos.

Semántica

Las palabras clave de operador **alt**, **par**, **loop**, **opt** y **exc** que en la representación gráfica se colocan en la esquina superior izquierda designan, respectivamente, composición alternativa, composición paralela, iteración, región facultativa y excepción. En la forma gráfica un marco engloba los operandos y las líneas de trazo designan separadores de operandos.

El operador **alt** define ejecuciones alternativas de secciones de MSC. Esto significa que si se consideran varias secciones de MSC como alternativas, únicamente se ejecutará una de ellas. En el caso en que secciones MSC alternativas tengan un preámbulo común, la elección de la sección MSC que se ejecutará se realiza después de la ejecución del preámbulo común.

El operador **par** define la ejecución paralela de secciones de MSC. Esto significa que se ejecutarán todos los eventos contenidos en las secciones de MSC paralelas, con la única limitación de que deberá mantenerse el orden de eventos dentro de cada sección.

El constructivo **loop** puede tener varias formas. La forma más básica es "**loop** <n,m>" donde n y m son número naturales. Esto significa que el operando puede ejecutarse n veces como mínimo y m veces como máximo. Los números naturales pueden sustituirse por la palabra clave **inf** como "**loop** <n,**inf**>". Esto significa que se ejecutará el bucle al menos n veces. Si se omite el segundo operando como en "**loop** <n>", se interpreta como "**loop** <n,n>". En consecuencia "**loop <inf>**" representa un bucle infinito. Si se omiten los límites del bucle, como en **loop**, se interpretará como "**loop** <1,**inf**>". Si el primer operando es mayor que el segundo, el bucle no se ejecutará ninguna vez.

El operador **opt** es el mismo que una alternativa cuando el segundo operando es el MSC vacío.

El operador **exc** constituye una forma compacta para describir casos excepcionales en los MSC. El significado del operador es que o bien los eventos interiores al <exc inline expression symbol> se han ejecutado y entonces el MSC ha concluido o que se han ejecutado los eventos que siguen al <exc inline expression symbol>. Por consiguiente, puede contemplarse el operador **exc** como una forma alternativa en la que el segundo operando es todo el resto del MSC.

En la representación textual, la <inline gate interface> facultativa define los mensajes que entran o salen de la expresión en línea a través de puertas. Mediante la identificación del nombre de mensaje y la identificación del nombre de la puerta facultativa, la <inline gate interface> define, asimismo, la conexión de mensaje directa entre dos expresiones en línea.

5.4 Referencia de MSC

Se utilizan referencias de MSC para referirse a otros MSC del documento MSC. Las referencias de MSC son objetos del tipo dado por el MSC referenciado.

Las referencias de MSC pueden referirse no solamente a un MSC aislado sino también a expresiones de referencia de MSC. Las expresiones de referencia de MSC son expresiones de MSC textuales construidas a partir de los operadores **alt**, **par**, **seq**, **loop**, **opt**, **exc** y **subst** y referencias de MSC.

Los operadores **alt**, **par**, **loop**, **opt** y **exc** se describen en 5.3. El operador **seq** designa la operación de secuenciación débil en la que únicamente se ordenan eventos del mismo caso.

La operación **subst** es una substitución de conceptos dentro del MSC referenciado. Se sustituyen nombres de mensajes por nombres de mensajes, nombres de casos por nombres de casos y nombres de MSC por nombres de MSC.

Las puertas reales de la referencia de MSC pueden conectarse con los constructivos correspondientes de MSC de cierre. Por constructivos correspondientes entendemos que una puerta de mensaje real puede conectarse con otra puerta de mensaje real o con un caso o con una definición de puerta de mensaje del MSC de cierre. Además, una puerta de orden real puede conectarse con otra puerta de orden real o con un evento ordenable o con una definición de puerta de orden.

Gramática textual concreta

<shared msc reference> ::=
reference [<msc reference identification>**:**] <msc ref expr>
<shared> [<reference gate interface>]

<msc reference> ::=
reference [<msc reference identification>**:**] <msc ref expr>
[<reference gate interface>]

<msc reference identification> ::=
<msc reference name>

<msc ref expr> ::=
<msc ref par expr> { **alt** <msc ref par expr> }*

<msc ref par expr> ::=
<msc ref seq expr> { **par** <msc ref seq expr> }*

<msc ref seq expr> ::=
<msc ref exc expr> { **seq** <msc ref exc expr> }*

<msc ref exc expr> ::=
[**exc**] <msc ref opt expr>

<msc ref opt expr> ::=
[**opt**] <msc ref loop expr>

<msc ref loop expr> ::=
[**loop**] [<loop boundary>]
{ **empty** | <msc name> [<parameter substitution>] | (<msc ref expr>) }

<parameter substitution> ::=
subst <substitution list>

<substitution list> ::=
<substitution> [, <substitution list>]

<substitution> ::=
<replace message> | <replace instance> | <replace msc>

<replace message> ::=
[**msg**] <message name> **by** <message name>

<replace instance> ::=
[**inst**] <instance name> **by** <instance name>

<replace msc> ::=
[**msc**] {**empty** | <msc name> } **by** {**empty** | <msc name> }

<reference gate interface> ::=
{ <end> **gate** <ref gate> }*

<ref gate> ::=
<actual out gate> | <actual in gate> | <actual order out gate> | <actual order in gate>

Si hay ambigüedades con respecto a los nombres en las sustituciones, deberán eliminarse utilizando las palabras claves facultativas **msg**, **inst** y **msc**.

La sustitución de un nombre de MSC debe compartir la misma interfaz de puerta a la que reemplaza.

Debe unirse una referencia de msc a cada caso presente en el diagrama de cierre contenido en el MSC al que se refiere al caso msc. Si dos diagramas referenciados por dos referencias de msc dentro de un diagrama de cierre comparten los mismos casos, éstos deben también aparecer en el diagrama de cierre.

Puede unirse una referencia de msc a casos no contenidos en el diagrama referenciado.

La interfaz de la referencia de MSC debe concordar con la interfaz de los MSC referenciados en la expresión, es decir cualquiera de las puertas unidas a la referencia debe tener una definición de puerta correspondiente en los MSC referenciados. La correspondencia viene dada por el sentido y el nombre del mensaje unido con la puerta y, si está presente, por el nombre de la puerta que es único dentro de la expresión referenciada.

En el caso en que <msc ref expr> conste de una expresión de operador textual en vez de un simple <msc name> y cuando más de una referencia de MSC se refiera al mismo MSC, deberá emplearse el <msc reference name> facultativo de <msc reference identification> para direccionar una referencia de MSC en la definición del mensaje (véase 4.3).

Gramática gráfica concreta

<msc reference area>::=

<msc reference symbol>
contains { <msc ref expr> [<actual gate area>*] } ***set***
is attached to { <instance axis symbol>* } ***set***
is attached to { <actual gate area>* } ***set***

<msc reference symbol>::=

T1009460-96/d28

La <msc reference area> puede unirse a uno o más casos. Si una <msc reference area> compartida atraviesa un <instance axis symbol> no implicado en esta referencia de MSC se dibuja a través el <instance axis symbol>.

Las referencias de MSC no deben unirse a casos descompuestos.

Las referencias de MSC no deben referirse directa ni indirectamente a sus MSC circundantes (recurrencia).

Un MSC que contenga referencias y sustituciones es ilegal, si tras una ampliación repetida de las referencias y una aplicación de las sustituciones, resulta un MSC ilegal.

Semántica

Cada MSC puede contemplarse como definición de un tipo MSC. Los tipos MSC pueden utilizarse en otros tipos MSC a través de referencias de MSC.

Un tipo MSC puede unirse a su entorno mediante puertas-mensaje. Se utilizan las puertas para definir los puntos de conexión en el caso en que un tipo MSC se utilice en otro tipo. Pueden asociarse a los puntos de conexión identificadores de puerta en forma de nombres.

En general, la referencia de MSC puede referirse a una expresión de MSC textual modificada posiblemente por una sustitución. En el caso elemental en que una expresión de MSC conste de un nombre de MSC, la referencia de MSC apunta a una definición de tipo de MSC correspondiente. La correspondencia viene dada por el nombre del MSC que es único dentro del documento del MSC. El resultado de una referencia con sustitución es que los reemplazos de la sustitución modificarán la definición del MSC referenciado. **Subst** x **by** y significa que se reemplaza x por y. Todas las sustituciones de una <substitution list> se consideran aplicadas en paralelo. Es decir el orden en el que se realizan las sustituciones es indiferente.

Si una definición de MSC al que se aplican sustituciones, contiene referencias de MSC, deberá también aplicarse la sustitución a las definiciones de MSC correspondientes a estas referencias de MSC.

En la representación textual, la <reference gate interface> facultativa define los mensajes que entran o salen de la referencia de MSC a través de puertas. La <reference gate interface> define también la conexión de mensaje directo entre dos referencias de MSC mediante la identificación del nombre de mensaje y la identificación, facultativa, del nombre de la puerta.

Una referencia de MSC con la palabra clave **empty** se refiere a un MSC que no tiene ni eventos ni casos.

5.5 MSC de alto nivel (HMSC)

Los MSC de alto nivel proporcionan medios para definir gráficamente cómo puede combinarse un conjunto de MSC. Un HMSC es un gráfico dirigido en el que cada nodo puede ser:

- un símbolo de arranque (en cada HMSC únicamente hay un símbolo de arranque);
- un símbolo de fin;
- una referencia de MSC;
- una condición;
- un punto de conexión, o
- un marco paralelo.

Las líneas de flujo conectan los nodos del HMSC e indican el secuenciamiento posible entre los nodos del HMSC. Las líneas de flujo entrantes se conectan siempre al borde superior de los símbolos del nudo en tanto que las líneas de flujo salientes se conectan al borde inferior. Si hay más de una línea de flujo saliente desde un nodo, ello indica una alternativa.

Pueden utilizarse referencias de MSC para referenciar un solo MSC o varios MSC empleando una expresión de MSC textual.

Pueden utilizarse condiciones en los HMSC para indicar estados de sistema globales e imponer limitaciones a los MSC referenciados en el HMSC.

Los marcos paralelos contienen uno o más HMSC pequeños e indican que los HMSC pequeños son los operandos de un operador paralelo, es decir que pueden intercalarse los eventos de diferentes HMSC pequeños.

Se introducen los puntos de conexión para simplificar la representación de los HMSC pero carecen de significado semántico.

Gramática textual concreta

<msc expression> ::=
<start> <node expression> *

<start> ::=
<label name> {**alt** <label name> }* <end>

<node expression> ::=
<label name> **:** { <node> **seq** (<label name> { **alt** <label name> }*) | **end** }<end>

<node> ::=
(<msc ref expr>)
| **empty** | <msc name>
| <par expression>
| **condition** <condition name list>
| **connect**

<par expression> ::=
expr <msc expression> **endexpr** { **par expr** <msc expression> **endexpr** }*

Semántica estática

En el HMSC deben definirse todos los <label name> referenciados en <start> o <node expression>, es decir deben aparecer como "<label name>**:**" en una <node expression>. Desde <start> debe poderse llegar a cada nodo del gráfico HMSC, esto es, el gráfico debe estar conectado.

Gramática gráfica concreta

<mscexpr area> ::=
{ <start area> <node expression area>* <hmsc end area>* } ***set***

<start area> ::=
<hmsc start symbol> ***is followed by*** { <alt op area>$^{+}$ } ***set***

<hmsc start symbol> ::=

<hmsc end area> ::=

<hmsc end symbol> ***is attached to*** { <hmsc line symbol>$^{+}$} ***se***

<hmsc end symbol> ::=

<hmsc line symbol> ::=

<hmsc line symboll> | <hmsc line symbol2>

<hmsc line symboll> ::=

<hmsc line symbol2> ::=

<alt op area> ::=

<hmsc line symbol> ***is attached to*** { <node area> | <hmsc end symbol>

<node expression area> ::=

<node area> ***is followed by*** { < alt op area > $^{+}$} ***set***
is attached to { <hmsc line symbol> $^{+}$} ***set***

<node area> ::=

<hmsc reference area>
| <connection point symbol>
| <hmsc condition area>
| <par expr area>

<hmsc reference area> ::=

<msc reference symbol> ***contains*** <msc ref expr>

<connection point symbol> ::=

<hmsc condition area>::=

<condition symbol> ***contains*** <condition name list>

<par expr area>::=

<par frame symbol> ***contains*** { <mscexpr area>$^{+}$} ***set***

<par frame symbol>::=

<frame symbol>

Regla de dibujo

El <hmsc line symbol> puede ser una línea quebrada y tener cualquier sentido.

El que <hmsc start symbol> vaya seguido de la <alt op area> significa que los <hmsc line symbol> deben unirse a la esquina inferior del <hmsc start symbol> como se indica en la figura 5-2a, en el que dos líneas <hmsc line symbol> siguen a un <hmsc start symbol>:

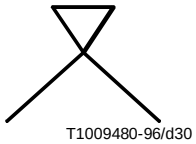


Figura 5-2a/Z.120 – Reglas de dibujo de HMSC

El que el <hmsc line symbol> esté unido a otro símbolo en la regla de producción de la <alt op area> significa que los <hmsc line symbol> deben unirse al borde superior del símbolo en cuestión. En los casos en que el símbolo sea un <msc reference symbol> y un <hmsc end symbol>, ello se ilustra en la figura 5-2b.

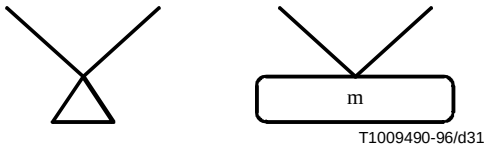


Figura 5-2b/Z.120 – Reglas de dibujo de HMSC

El que la <node area> vaya seguida de una <alt op area> en la regla de producción de la <node expression area> significa que los <hmsc line symbol> deben unirse al borde inferior del símbolo de la <node area>, como se indica en la figura 5-2c que muestra cómo un <msc reference symbol> va seguido de una <alt op area>:

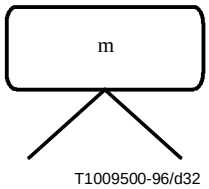


Figura 5-2c/Z.120 – Reglas de dibujo de HMSC

Semántica estática

La composición de MSC descrita mediante HMSC puede salvaguardarse mediante condiciones aplicadas a los HMSC. Pueden utilizarse estas condiciones para indicar estados de sistema globales. Obsérvese que únicamente pueden emplearse condiciones globales para limitar la composición de los MSC. Se considera que todas las condiciones de los HMSC son condiciones globales. En un MSC simple (MSC descrito mediante casos, mensajes, etc.) únicamente se consideran globales aquellas condiciones compartidas por todos los casos del MSC las cuales pueden utilizarse para limitar la composición en los HMSC. Se considera que las condiciones no globales de los MSC simples son comentarios.

Cada MSC simple, HMSC, marco paralelo (de un HMSC) y expresión MSC (en una referencia de MSC) tiene un conjunto de condiciones iniciales y un conjunto de condiciones finales que cumplen las siguientes reglas:

El conjunto de condiciones iniciales de un MSC simple (es decir un MSC no descrito por un diagrama de HMSC), se define como sigue:

- Si el MSC contiene una condición global que precede a todo salvo las cabeceras de caso, se define el conjunto de condiciones iniciales como los <condition name> definidos por esta condición.
- Si el MSC carece de esa condición global, se define el conjunto de condiciones iniciales como el conjunto de todos los <condition name> posibles. Esto se aplica también al MSC 'vacío'.

El conjunto de condiciones finales de un MSC simple se define como sigue:

- Si el MSC contiene una condición global que únicamente va seguida de los finales de caso, se define el conjunto de condiciones finales como los <condition name> definidos por esa condición.
- Si el MSC carece de esa condición global, se define el conjunto de condiciones finales como el conjunto de todos los <condition name> posibles. Esto se aplica también al MSC 'vacío'.

El conjunto de condiciones iniciales de un HMSC se define como sigue:

- Si el <hmsc start symbol> va seguido de uno o más <condition symbol>, se define el conjunto de condiciones iniciales como la intersección de los conjuntos de <condition name> definidos por cada uno de los <condition symbols>. Obsérvese que si una rama saliente del <start symbol> comienza con más de un <condition symbol> para calcular las condiciones iniciales del HMSC únicamente se tiene en cuenta el primero.
- Si el <start symbol> no va seguido inmediatamente de un <condition symbol>, se define el conjunto de condiciones iniciales como el conjunto completo de todos los <condition name> posibles.

El conjunto de condiciones finales de un HMSC, se define como sigue:

- Si el HMSC contiene uno o más <hmsc end symbol> que van inmediatamente precedidos de <condition symbol>, se define el conjunto de condiciones finales como la intersección de los conjuntos <condition name> definidos por cada uno de los <condition symbols>.
- Si ninguno de los <hmsc end symbol> del HMSC va precedido inmediatamente de un <condition symbol>, se define el conjunto de condiciones finales como el conjunto completo de todos los <condition name> posibles.

Se define el conjunto de condiciones iniciales de una <msc ref expression> (esto es la expresión asociada a una <msc reference>), como sigue (donde m1 es el nombre de un MSC y e1 y e2 son expresiones de MSC):

- El conjunto de condiciones iniciales de la expresión 'm1' es el conjunto de condiciones iniciales de m1 del MSC.
- El conjunto de condiciones iniciales de 'e1 seq e2' es el conjunto de condiciones iniciales de e1.
- El conjunto de condiciones iniciales de 'e1 alt e2' es la intersección de los conjuntos de condiciones iniciales de e1 y e2.
- El conjunto de condiciones iniciales de 'e1 par e2' es la intersección de los conjuntos de condiciones iniciales de e1 y e2.
- El conjunto de condiciones iniciales de 'opt e1' es el conjunto de condiciones iniciales de e1.
- El conjunto de condiciones iniciales de 'loop e1' es el conjunto de condiciones iniciales de e1.

El conjunto de condiciones finales de una expresión de MSC en una referencia MSC, se define como sigue (siendo m1 el nombre de un MSC y e1 y e2 expresiones de MSC):

- El conjunto de condiciones finales de la expresión 'm1' es el conjunto de condiciones finales del m1 de MSC.
- El conjunto de condiciones finales de 'e1 seq e2' es el conjunto de condiciones finales de e2.
- El conjunto de condiciones finales de 'e1 alt e2' es la intersección de los conjuntos de condiciones finales de e1 y e2.
- El conjunto de condiciones finales de 'e1 par e2' es la intersección de los conjuntos de condiciones finales de e1 y e2.
- El conjunto de condiciones finales de 'opt e1' es el conjunto de condiciones finales de e1.
- El conjunto de condiciones finales de 'loop e1' es el conjunto de condiciones finales de e1.

Las condiciones iniciales y finales de la <par expr area> en los HMSC se definen de la misma forma que para las expresiones **par**:

- El conjunto de condiciones iniciales de la <par expr area> es la intersección del conjunto de condiciones iniciales de los operandos.
- El conjunto de condiciones finales de la <par expr area> es la intersección del conjunto de condiciones finales de los operandos.

En los HMSC hay cuatro restricciones estáticas relacionadas con las condiciones:

- Si una <msc reference> va inmediatamente precedida de un <condition symbol>, con un conjunto asociado de <condition name>, este conjunto debe ser un subconjunto del conjunto de condiciones iniciales de la <msc ref expression> asociada con la <msc reference>.
- Si una <msc reference> va inmediatamente seguida de un <condition symbol>, con un conjunto asociado de <condition name>, este conjunto debe ser un subconjunto del conjunto de condiciones finales de la <msc ref expression> asociada con la <msc reference>.

- Si una <par expr area> va inmediatamente precedida de un <condition symbol>, con un conjunto asociado de <condition name>, este conjunto debe ser un subconjunto del conjunto de condiciones iniciales de la <par expr area>.
- Si una <par expr area> va inmediatamente seguida de un <condition symbol>, con un conjunto asociado de <condition name>, este conjunto debe ser un subconjunto del conjunto de condiciones finales de la <par expr area>.

Semántica

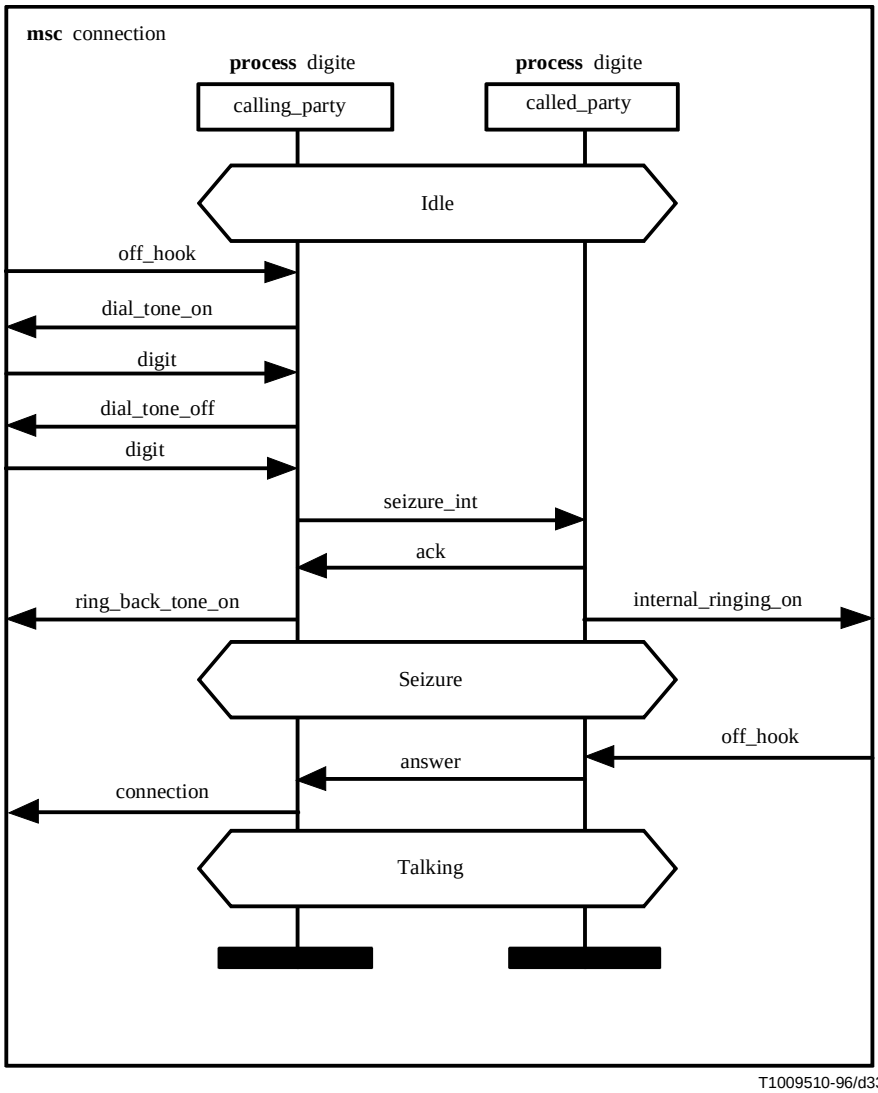
El gráfico que describe la composición de MSC dentro de un HMSC se interpreta de una forma operacional, como sigue: la ejecución comienza en el <hmsc start symbol>. A continuación prosigue con un nodo posterior a uno de los bordes de salida del símbolo. Se considera que estos nodos son los operandos de un operador **alt** (véase 5.3). Tras la ejecución del nodo seleccionado, se repite el proceso de selección y ejecución para los bordes de salida del nodo seleccionado. La ejecución de un nodo final implica la ejecución de la terminación del HMSC dado. La ejecución de una referencia de MSC se realiza según la descripción de 5.4. La ejecución de una condición o un punto de conexión es una operación vacía. La ejecución de un marco paralelo, en paralelo, de dicho marco como se describe en 5.3 para el operador **par**. El operador **seq** (véase 5.4) describe una ejecución secuencial de dos nodos relacionados mediante un borde.

6 Ejemplos de gráficos de secuencias de mensajes

Esta cláusula es informativa en vez de normativa.

6.1 Diagrama de flujo de mensajes normalizado

En este ejemplo se muestra una conexión simplificada dentro de un sistema de conmutación. Se ilustran los constructivos de MSC más básicos: casos (procesos), entorno, mensajes, condiciones globales.

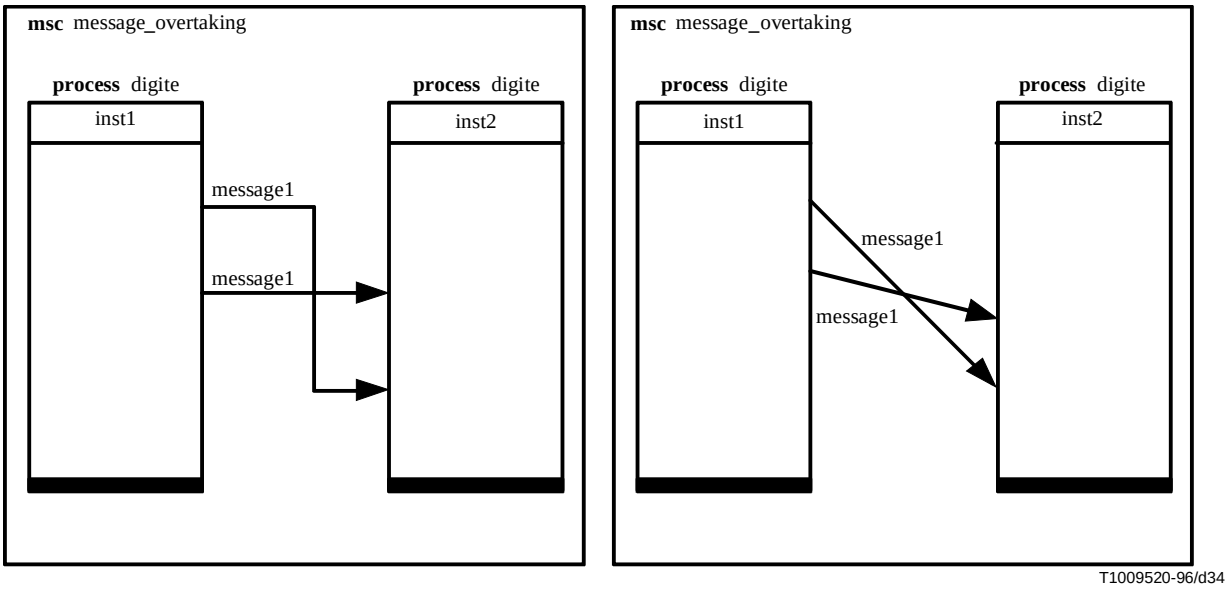


T1009510-96/d33

```
msc connection; inst calling_party: process digite, called_party: process digite;
        instance calling_party: process digite;
            condition Idle shared all;
            in off_hook from env;
            out dial_tone_on to env;
            in digit from env;
            out dial_tone_off to env;
            in digit from env;
            out seizure_int to called_party;
            in ack from called_party;
            out ring_back_tone_on to env;
            condition Seizure shared all;
            in answer from called_party;
            out connection to env;
            condition Talking shared all;
        endinstance;
        instance called_party: process digite;
            condition Idle shared all;
            in seizure_int from calling_party;
            out ack to calling_party;
            out internal_ringing_on to env;
            condition Seizure shared all;
            in off_hook from env;
            out answer to calling_party;
            condition Talking shared all;
        endinstance;
endmsc;
```

6.2 Adelantamiento de mensajes

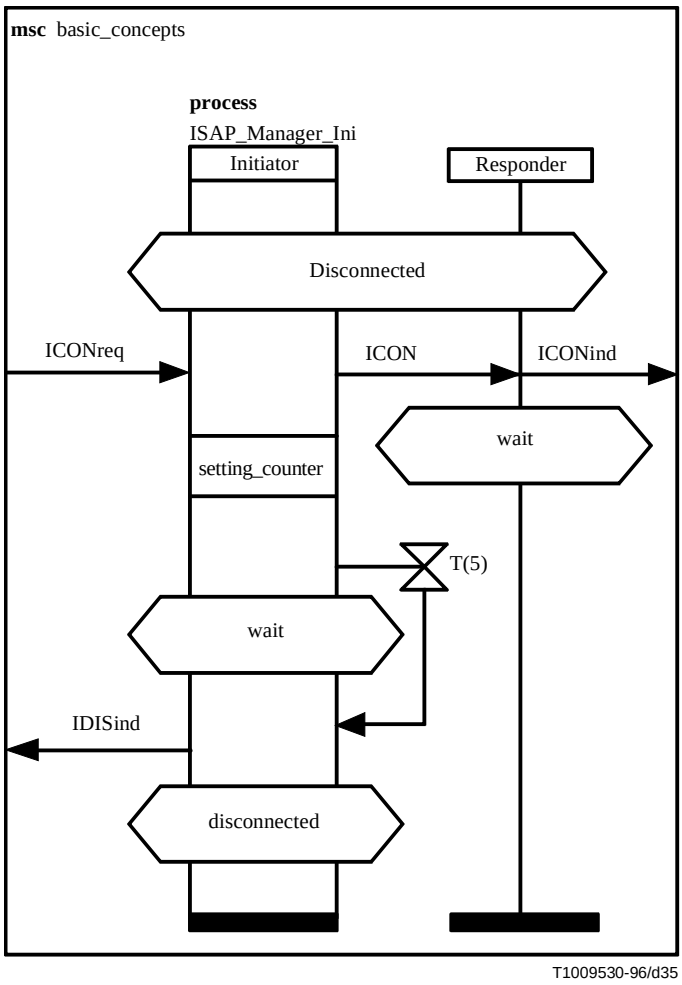
En este ejemplo se muestra el adelantamiento de dos mensajes con el mismo nombre: "message 1". En la representación textual, se emplean los nombres de caso de mensaje (a, b) para una correspondencia unívoca entre entrada y salida de mensaje. En la representación gráfica, los mensajes se representan mediante flechas horizontales, una doblada para indicar el adelantamiento o mediante flechas inclinadas que se cruzan.




```
msc message_overtaking; inst inst1, inst2;
        instance inst1: process digite;
            out message1, a to inst2;
            out message1, b to inst2;
        endinstance;
        instance inst2: process digite;
            in message1, b from inst1;
            in message1, a from inst1;
        endinstance;
endmsc;
```

6.3 Conceptos básicos de MSC

Este ejemplo contiene los constructivos de MSC básicos: casos, entorno, mensajes, condiciones, acciones y temporización. En la representación gráfica se utilizan los dos tipos de símbolos de caso: el renglón y la columna.



Sintaxis textual orientada a los casos

msc basic_concepts; **inst** Initiator: **process** ISAP_Manager_Ini, Responder;
instance Initiator: **process** ISAP_Manager_Ini;
condition Disconnected **shared all**;
in ICONreq **from env**;
out ICON **to** Responder;
action setting_counter;
set T(5);
condition wait **shared**;
timeout T;
out IDISind **to env**;
condition disconnected **shared**;
endinstance;
instance Responder;
condition Disconnected **shared all**;
in ICON **from** Initiator;
out ICONind **to env**;
condition wait **shared**;
endinstance;
endmsc;

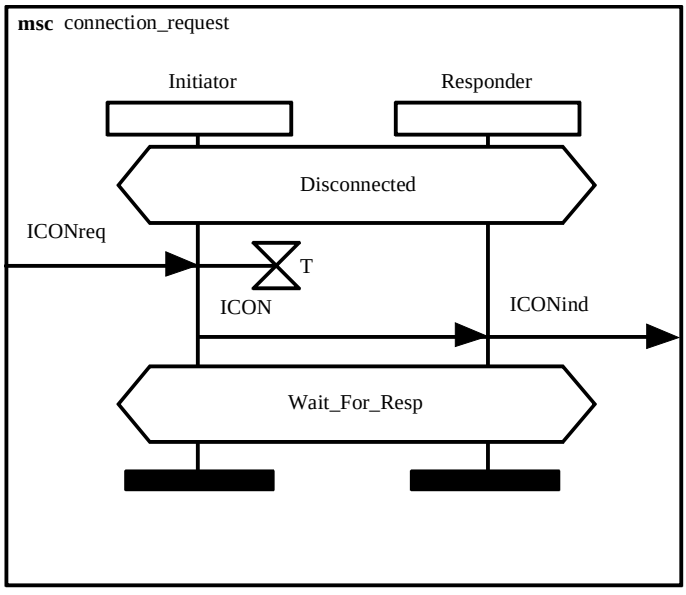
Sintaxis textual orientada a los eventos

```
msc basic_concepts; inst Initiator: process ISAP_Manager_Ini, Responder;
Initiator:      instance process ISAP_Manager_Ini;
Responder:      instance;
all:            condition Disconnected;
Initiator:      in ICONreq from env;
                out ICON to Responder;
Responder:      in ICON from Initiator;
                out ICONind to env;
                condition wait;
                endinstance;
Initiator:      action setting_counter;
                set T(5);
                condition wait;
                timeout T;
                out IDISind to env;
                condition disconnected;
                endinstance;
endmsc;
```

6.4 Composición/descomposición de MSC

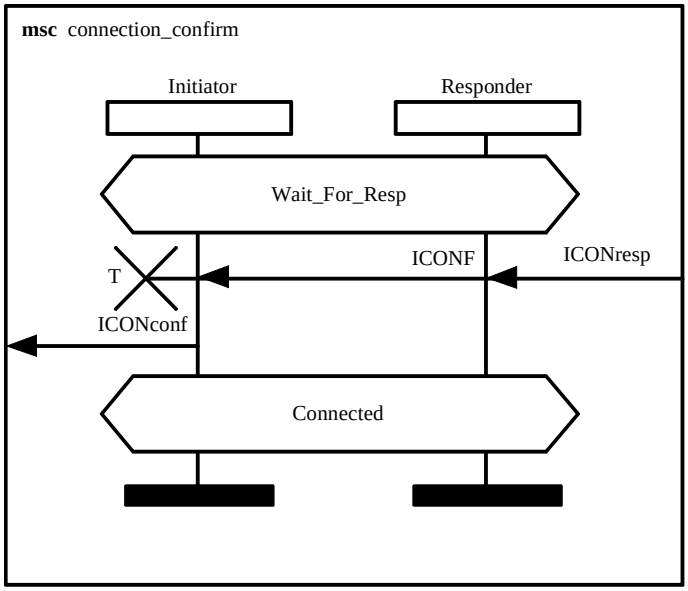
En este ejemplo se ilustra la composición de MSC mediante condiciones globales. La condición global final "Wait For Resp" de una petición de conexión MSC es idéntica a la condición global inicial de la confirmación de conexión de MSC. Por consiguiente, pueden componerse ambos MSC en la conexión de MSC resultante (ejemplo 6.5).

La composición se define mediante el "con_setup" del HMSC

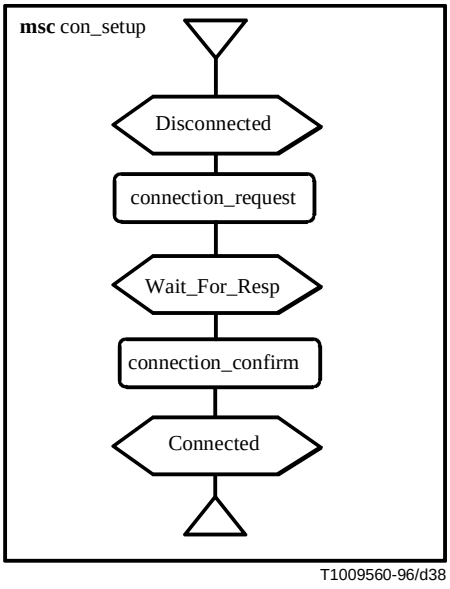


T1009540-96/d36

```
msc connection_request; inst Initiator, Responder;
        instance Initiator;
            condition Disconnected shared all;
            in ICONreq from env;
            set T;
            out ICON to Responder;
            condition Wait_For_Resp shared all;
        endinstance;
        instance Responder;
            condition Disconnected shared all;
            in ICON from Initiator;
            out ICONind to env;
            condition Wait_For_Resp shared all;
        endinstance;
endmsc;
```



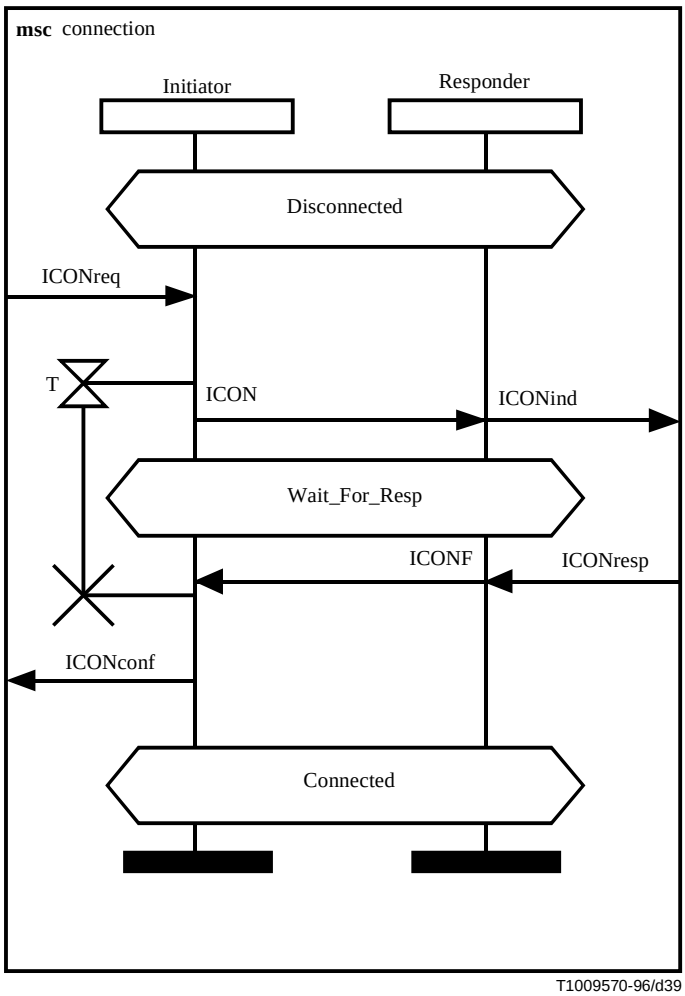
msc connection_confirm; **inst** Initiator, Responder;
instance Inititator;
condition Wait_For_Resp **shared all**;
in ICONF **from** Responder;
reset T;
out ICONconf **to env**;
condition Connected **shared all**;
endinstance;
instance Responder;
condition Wait_For_Resp **shared all**;
in ICONresp **from env**;
out ICONF **to** Initiator;
condition Connected **shared all**;
endinstance;
endmsc;



msc con_setup;
expr L1;
L1: **condition** Disconnected **seq** (L2);
L2: connection_request **seq** L3;
L3: **condition** Wait_For_Resp **seq** (L4);
L4: connection_confirm **seq** (L5);
L5: **condition** Connected **seq** (L6);
L6: **end**;
endmsc;

6.5 MSC con supervisión de tiempo

El MSC 'connection' de este ejemplo contiene una reiniciación de temporizador.

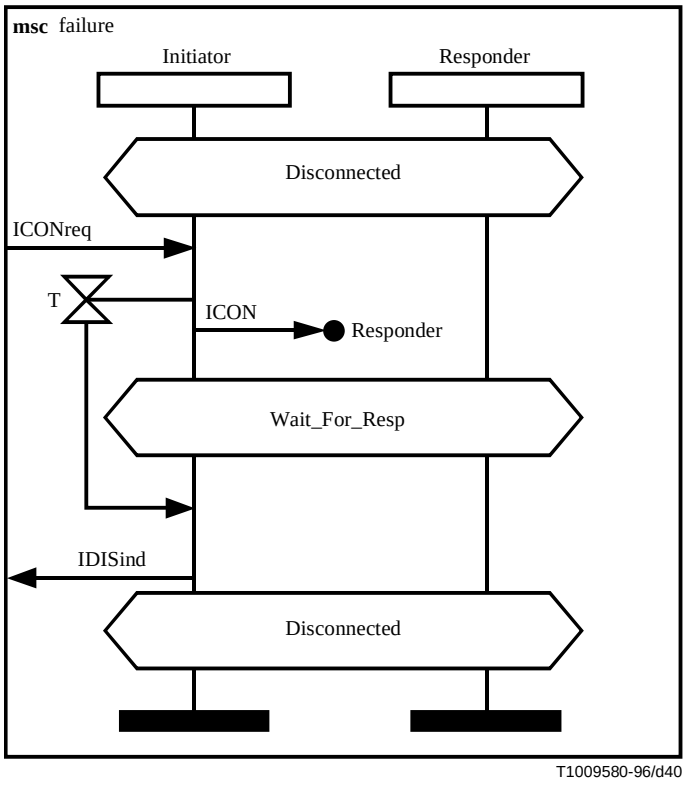


msc connection; **inst** Initiator, Responder;
instance Initiator;
condition Disconnected **shared all**;
in ICONreq **from env**;
set T;
out ICON **to** Responder;
condition Wait_For_Resp **shared all**;
in ICONF **from** Responder;
reset T;
out ICONconf **to env**;
condition Connected **shared all**;

endinstance;
instance Responder;
condition Disconnected **shared all**;
in ICON **from** Initiator;
out ICONind **to env**;
condition Wait_For_Resp **shared all**;
in ICONresp **from env**;
out ICONF **to** Initiator;
condition Connected **shared all**;
endinstance;
endmsc;

6.6 MSC con pérdida de mensajes

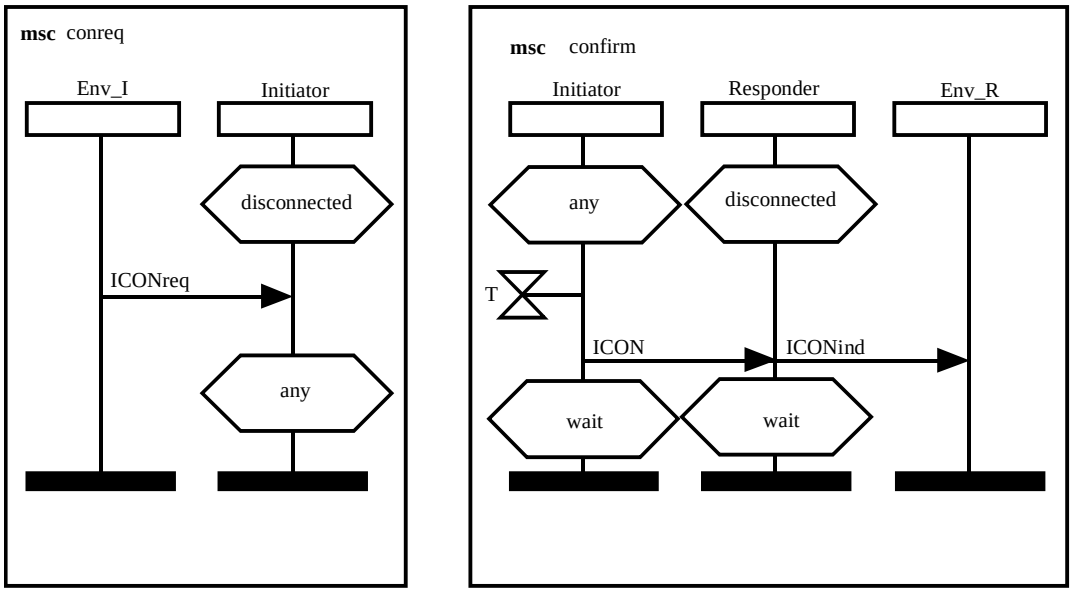
El MSC "failure" de este ejemplo contiene una expiración de temporizador debida a un mensaje perdido.



msc failure; **inst** Initiator, Responder;
instance Initiator;
condition Disconnected **shared all**;
in ICONreq **from env**;
set T;
out ICON **to lost** Responder;
condition Wait_For_Resp **shared all**;
timeout T;
out IDISind **to env**;
condition Disconnected **shared all**;
endinstance;
instance Responder;
condition Disconnected **shared all**;
in ICON **from** Initiator;
condition Wait_For_Resp **shared all**;
condition Disconnected **shared all**;
endinstance;
endmsc;

6.7 Condiciones locales

En este ejemplo se emplean condiciones locales que hacen referencia a un caso para indicar estados locales de este caso.

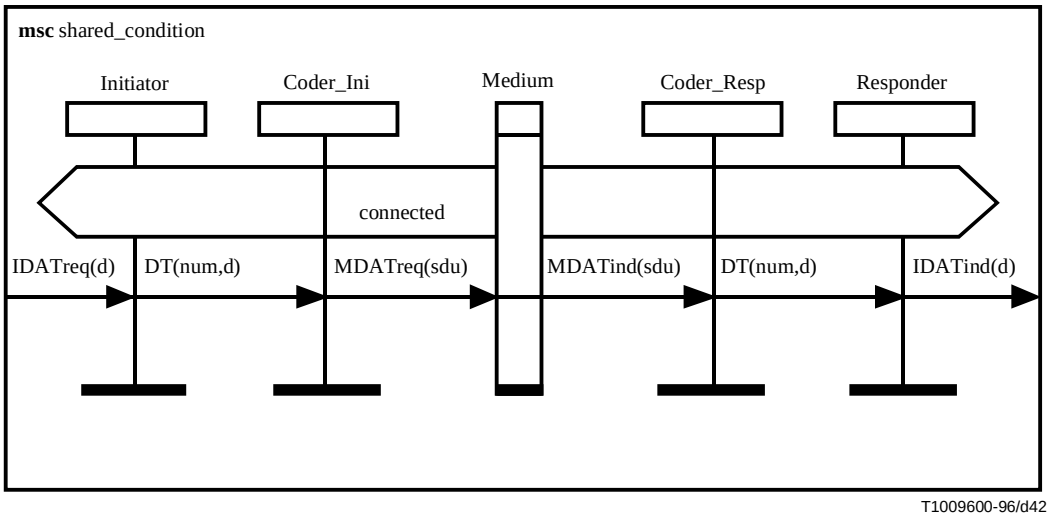



```
msc conreq; inst Env_I, Initiator;
        instance Env_I;
            out ICONreq to Initiator;
        endinstance;
        instance Initiator;
            condition disconnected shared;
            in ICONreq from Env_I;
            condition any shared;
        endinstance;
endmsc;


msc confirm; inst Initiator, Responder, Env_R;
        instance Initiator;
            condition any shared;
            set T;
            out ICON to Responder;
            condition wait shared;
        endinstance;
        instance Responder;
            condition disconnected;
            in ICON from Initiator;
            out ICONind to Env_R;
            condition wait shared;
        endinstance;
        instance Env_R;
            in ICONind from Responder;
        endinstance;
endmsc;
```

6.8 Condición compartida y mensajes con parámetros

Este ejemplo contiene la condición compartida "connected". Esta condición la comparten los casos "Initiator" y "Responder". No resultan afectados los casos "Coder_Ini", "Medium", "Coder_Resp". En la representación textual, la palabra clave **shared**, junto con una lista de casos, indica los casos a los que está unida la conexión.



T1009600-96/d42

msc shared_condition; **inst** Initiator, Coder_Ini, Medium, Coder_Resp, Responder;

instance Initiator;

condition connected **shared** Responder;

in IDATreq(d) **from env**;

out DT(num,d) **to** Coder_Ini;

endinstance;

instance Coder_Ini;

in DT(num,d) **from** Initiator;

out MDATreq(sdu) **to** Medium;

endinstance;

instance Medium;

in MDATreq(sdu) **from** Coder_Ini;

out MDATind(sdu) **to** Coder_Resp;

endinstance;

instance Coder_Resp;

in MDATind(sdu) **from** Medium;

out DT(num,d) **to** Responder;

endinstance;

instance Responder;

condition connected **shared** Initiator;

in DT(num,d) **from** Coder_Resp;

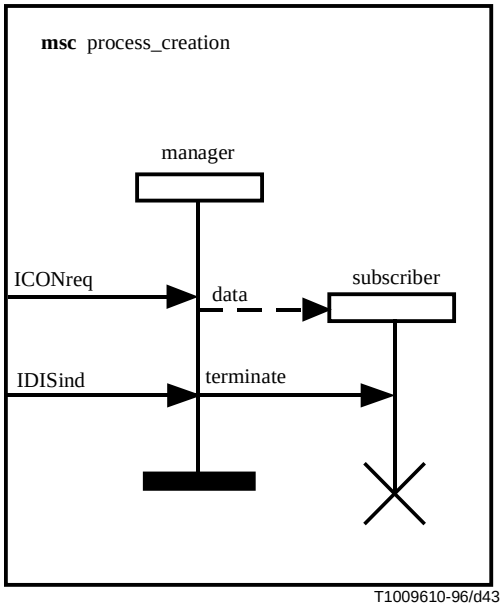
out IDATind(d) **to env**;

endinstance;

endmsc;

6.9 Creación y terminación de procesos

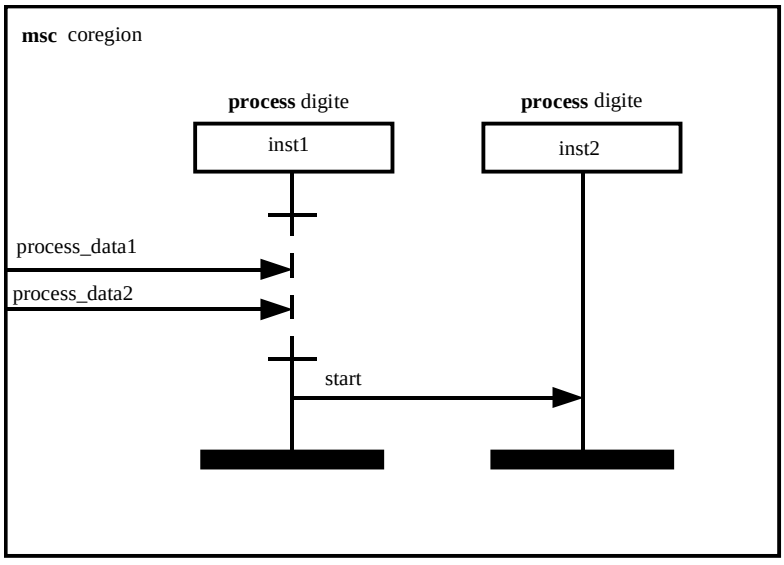
Este ejemplo ilustra la creación dinámica del caso "subscriber" ocasionada por una petición de conexión y su terminación correspondiente debida a una petición de desconexión.



msc process_creation; **inst** manager, subscriber;
instance manager;
in ICONreq **from env**;
create subscriber(data);
in IDISind **from env**;
out terminate **to** subscriber;
endinstance;
instance subscriber;
in terminate **from** manager;
stop;
endinstance;
endmsc;

6.10 Corregión

Este ejemplo muestra una región concurrente, que indicará que el consumo de "process_data1" y el consumo de "process_data2" no están ordenados temporalmente, es decir, "process_data1" se puede consumir antes de "process_data2" o al revés.

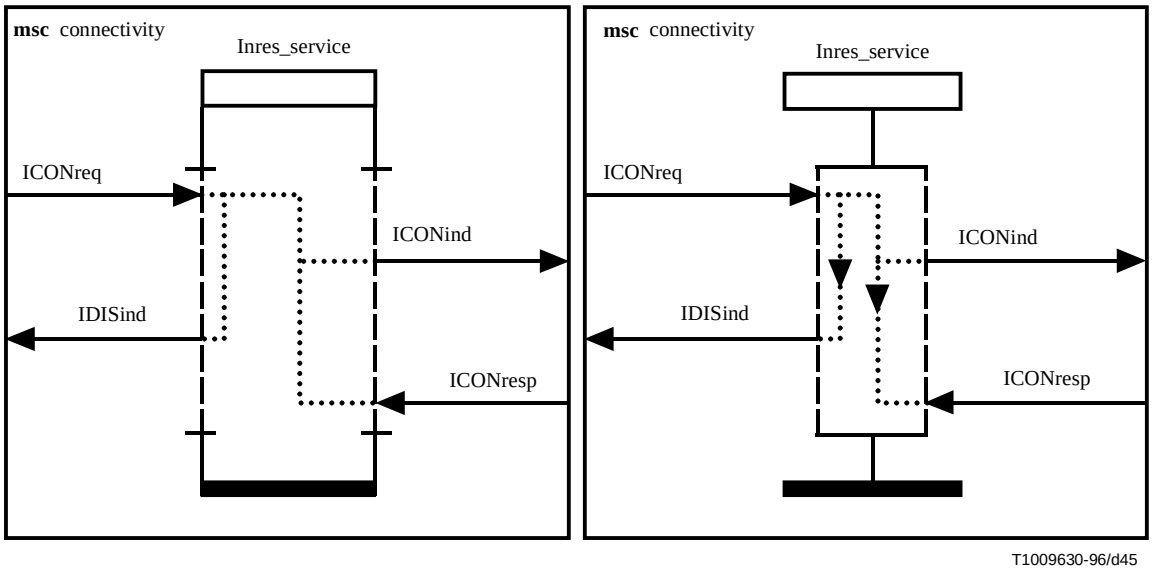


```
msc coregion; inst inst1, inst2;
        instance inst1: process digite;
            concurrent;
                in process_data1 from env;
                in process_data2 from env;
            endconcurrent;
            out start to inst2;
        endinstance;
        instance inst2: process digite;
            in start from inst1;
        endinstance;
endmsc;
```

6.11 Ordenación generalizada en una corregión

Este ejemplo muestra una ordenación generalizada dentro de una corregión mediante "connections", es decir se describe la ordenación mediante las conexiones que relacionan los eventos dentro de la corrección. Dentro de la "connectivity" del MSC, se define la siguiente ordenación: ICONreq <ICONind < ICONresp, ICONreq < IDISind.

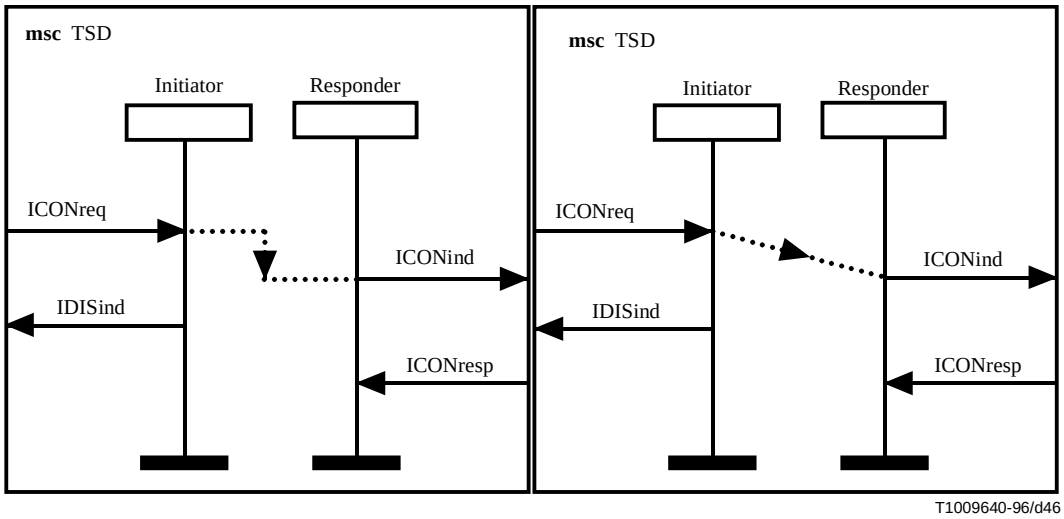
Se muestra la situación en la que IDISind está desordenada respecto de ICONind e ICONresp.




```
msc connectivity; inst Inres_service;
        instance Inres_service;
            concurrent;
                in ICONreq from env before Label1, Label2;
                Label1 out ICONind to env before Label3;
                Label2 out IDISind to env;
                Label3 in ICONresp from env;
            endconcurrent;
        endinstance;
endmsc;
```

6.12 Ordenación generalizada entre casos diferentes

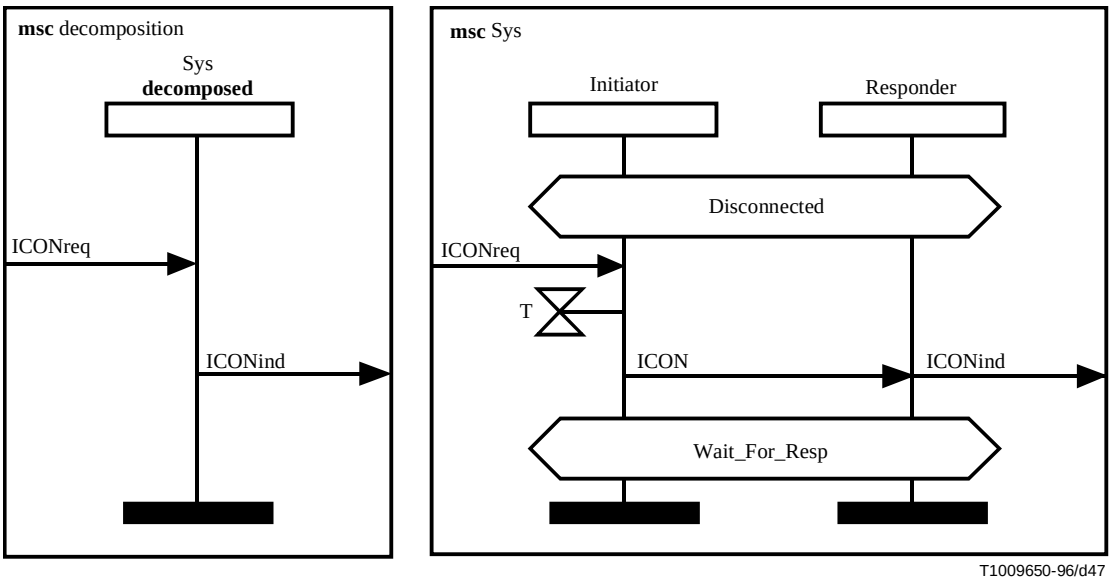
En este ejemplo se muestra la utilización de constructivos de sincronización a partir de los diagramas de secuencia temporal para describir una ordenación generalizada entre casos distintos. La línea (doblada o inclinada hacia abajo) entre la entrada de mensaje "ICONreq" y la salida de mensaje "ICONind", designa la ordenación ICONreq < ICONind.



msc TSD; **inst** Initiator, Responder;
instance Initiator;
in ICONreq **from env before** Label1;
out IDISind **to env**;
endinstance;
instance Responder;
Label1 **out** ICONind **to env**;
in ICONresp **from env**;
endinstance;
endmsc;

6.13 Descomposición de casos

Este ejemplo contiene el perfeccionamiento "Sys" de MSC. Este MSC está unido al caso "Sys" representando, de este modo, una descomposición de este caso.

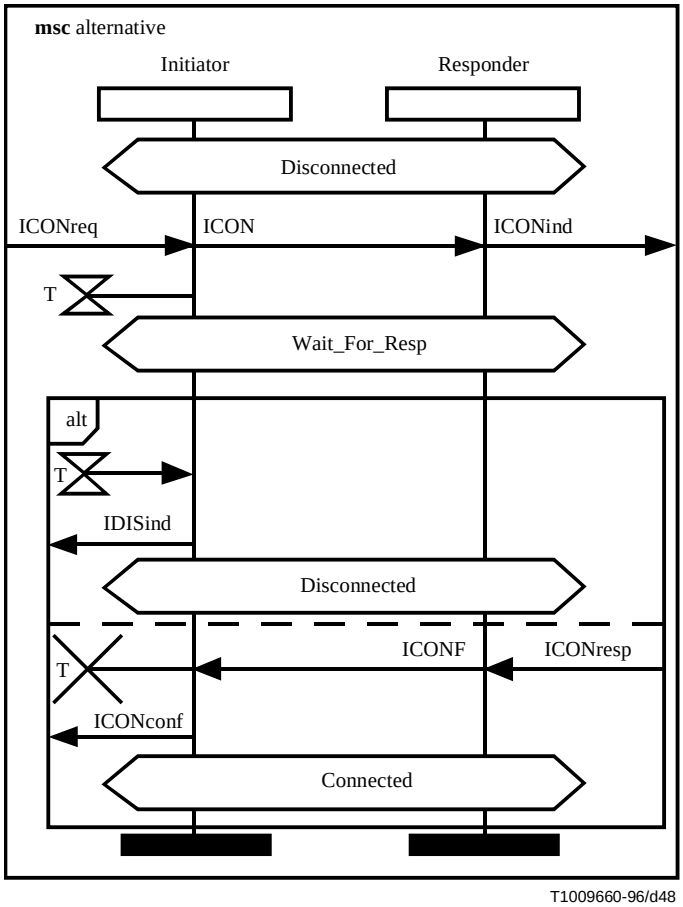


```
msc decomposition; inst Sys;
        instance Sys decomposed;
            in ICONreq from env;
            out ICONind to env;
        endinstance;
endmsc;

msc Sys; inst Initiator, Responder;
        instance Initiator;
            condition Disconnected shared all;
            in ICONreq from env;
            set T;
            out ICON to Responder;
            condition Wait_For_Resp shared all;
        endinstance;
        instance Responder;
            condition Disconnected shared all;
            in ICON from Initiator;
            out ICONind to env;
            condition Wait_For_Resp shared all;
        endinstance;
endmsc;
```

6.14 Expresión en línea con composición alternativa

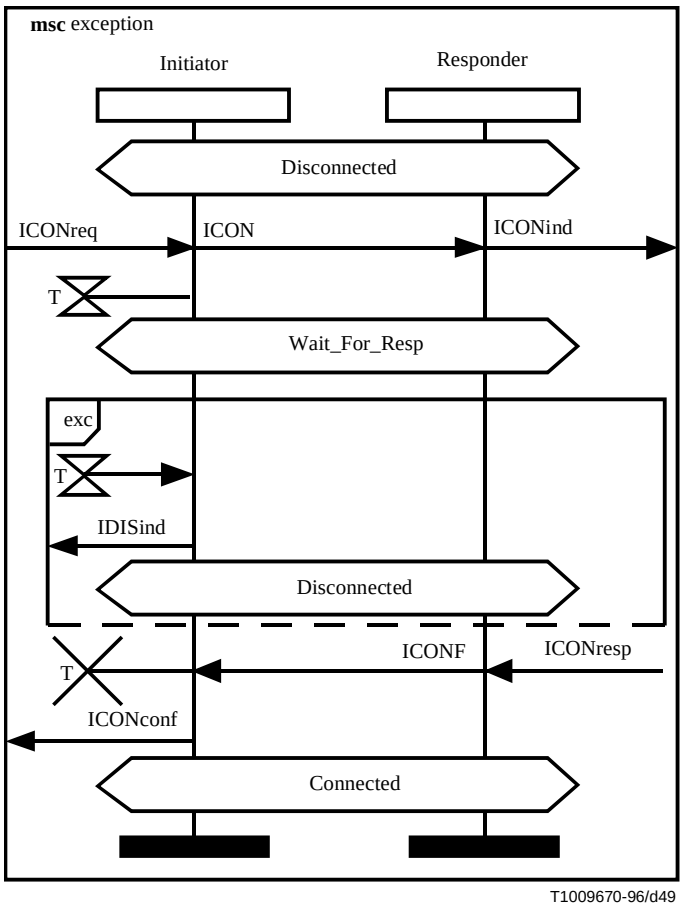
En este ejemplo se combina el caso de conexión satisfactoria con el caso de fallo dentro de un MSC mediante la expresión en línea alternativa ("alternative" de MSC). Dentro de la "exception" de MSC, se describe la misma situación mediante una expresión en línea de excepción equivalente.



```
msc alternative; inst Initiator, Responder;
        Initiator:      instance;
        Responder:      instance;
        all:            condition Disconnected;
        Initiator:      in ICONreq from env;
                        out ICON to Responder;
                        set T;
        Responder:      in ICON from Initiator;
                        out ICONind to env;
        all:            condition Wait for Resp;
                        alt begin;
            Initiator:      timeout T;
                            out IDISind to env;
            all:            condition Disconnected;
                        alt;
            Responder:      in ICONresp from env;
                            out ICONF to Initiator;
            Initiator:      in ICONF from Responder;
                            reset T;
                            out ICONconf to env;
            all:            condition Connected;
                        alt end;
        Initiator:      endinstance;
        Responder:      endinstance;
endmsc;
```

El operador **exc** es el mismo que una alternativa en la que el segundo operando es todo el resto del MSC como se muestra en el siguiente ejemplo:

"Alternative" de MSC significa exactamente lo mismo que "exception" de MSC:



T1009670-96/d49

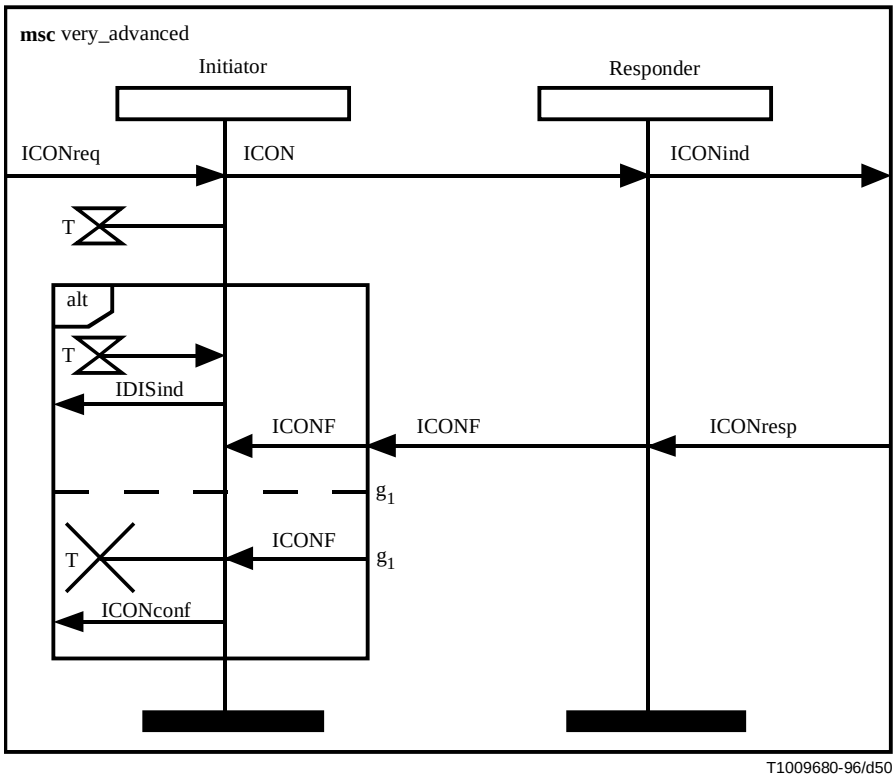
```
msc exception; inst Initiator, Responder;
        Initiator:      instance;
        Responder:      instance;
        all:            condition Disconnected;
        Initiator:      in ICONreq from env;
                        out ICON to Responder;
                        set T;
        Responder:      in ICON from Initiator;
                        out ICONind to env;
        all:            condition Wait for Resp;
                        exc begin;
            Initiator:      timeout T;
                            out IDISind to env;
            all:            condition Disconnected;
                        exc end;
        Responder:      in ICONresp from env;
                        out ICONF to Initiator;
        Initiator:      in ICONF from Responder;
                        reset T;
                        out ICONconf to env;
        all:            condition Connected;
        Initiator:      endinstance;
        Responder:      endinstance;
endmsc;
```

6.15 Expresión en línea con puertas

En este ejemplo se describe la situación de 6.14 de una forma ligeramente distinta: El mensaje "ICONF" procedente de "Responder" se conecta a través de puertas con la expresión en línea alternativa unida a "Initiator". El mensaje "ICONF" procedente de "Responder" se transfiere a través de la puerta g1 (en ambas alternativas) a "Initiator". Se describe la situación en la que "Initiator" espera una respuesta de "Responder". Ambos casos se combinan en el MSC "very_advanced":

a) el "Responder" no contesta a tiempo: la entrada de mensaje "ICONF" se descarta después de la temporización y la desconexión de "Initiator";

b) el "Responder" contesta a tiempo lo que produce una conexión satisfactoria.

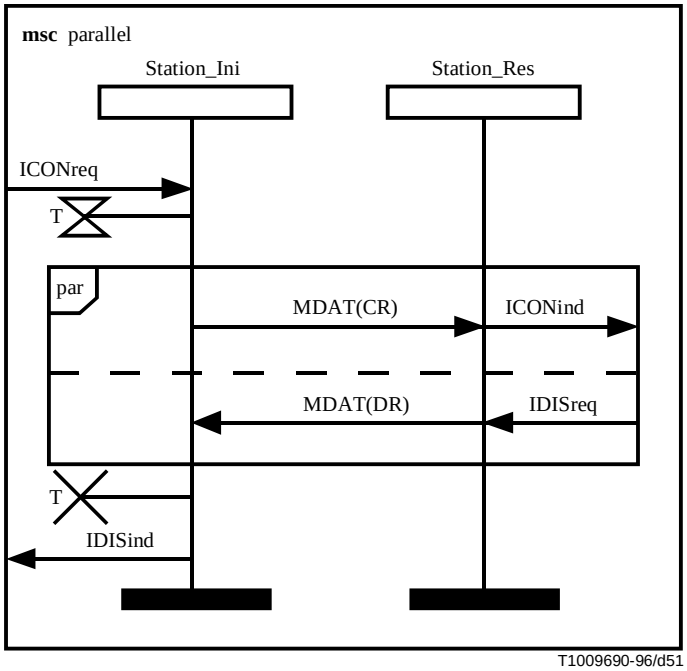


T1009680-96/d50

```
msc very_advanced; inst Initiator, Responder; gate out ICONreq to Initiator;
gate in ICONind from Responder; gate out ICONresp to Responder;
        Initiator:      instance;
        Responder:      instance;
        Initiator:      in ICONreq from env;
                        out ICON to Responder;
                        set T;
        Responder:      in ICON from Initiator;
                        out ICONind to env;
                        in ICONresp from env;
                        out ICONF to inline altref via g1;
        Initiator:      alt begin altref;
                            gate in IDISind from Initiator;
                            gate g1 out ICONF to Initiator;
                            external in ICONF from Responder;
                            timeout T;
                            out IDISind to env;
                            in ICONF from env via g1;
                        alt;
                            gate g1 out ICONF to Initiator;
                            gate in ICONconf from Initiator;
                            in ICONF from env via g1;
                            reset T;
                            out ICONconf to env;
                        alt end;
        Initiator:      endinstance;
        Responder:      endinstance;
endmsc;
```

6.16 Expresión en línea con composición paralela

En este ejemplo se muestra cómo una expresión en línea paralela describe la intercalación de una petición de conexión iniciada por "Station_Res", es decir "MDAT(CR)" seguida de "ICONind" con una petición de desconexión iniciada por el entorno, esto es, "IDISreq" seguida de "MDAT(DR)".

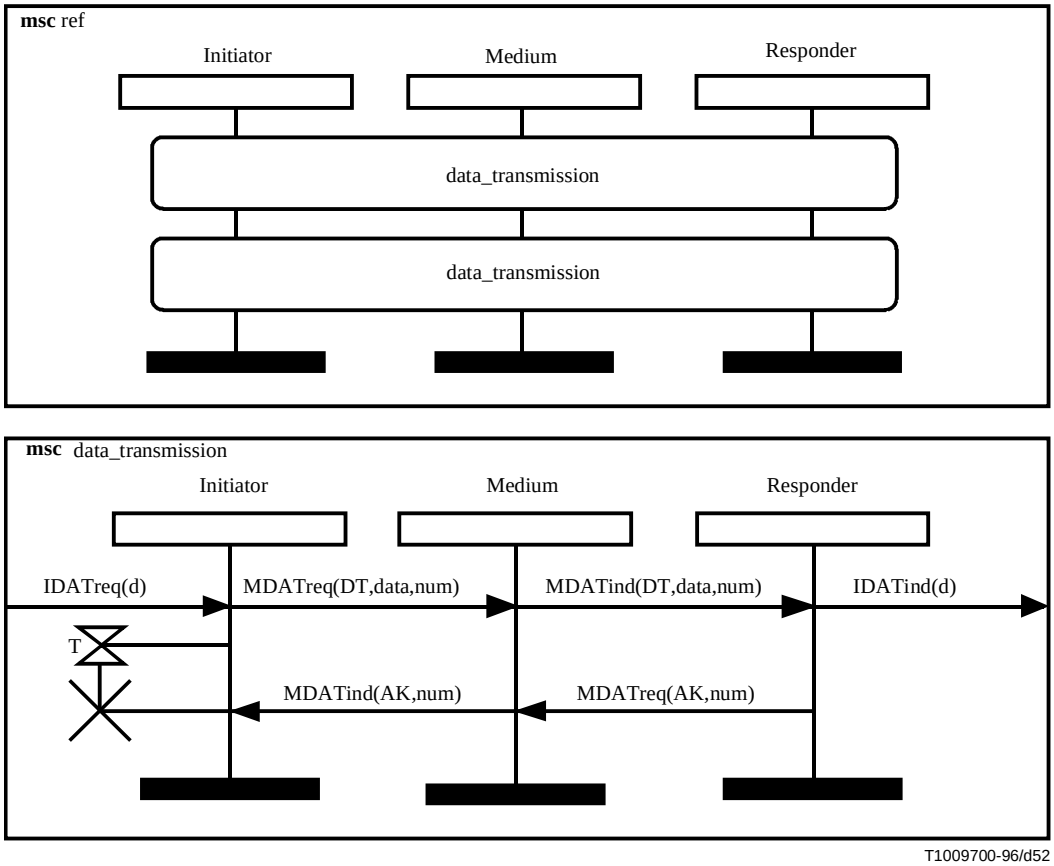


T1009690-96/d51

```
msc parallel; inst Station_Ini, Station_Res;
        Station_Ini:        instance;
                            in ICONreq from env;
                            set T;
        Station_Res:        instance;
        all:                par begin;
            Station_Ini:            out MDAT(CR) to Station_Res;
            Station_Res: in MDAT(CR) from Station_Ini;
                            out ICONind to env;
                            par;
            Station_Res: in IDISreq from env;
                            out MDAT(DR) to Station_Ini;
            Station_Ini:    in MDAT(DR) from Station_Res;
                            par end;
        Station_Res:        endinstance;
        Station_Ini: reset T;
                            out IDISind to env;
                            endinstance;
endmsc;
```

6.17 Referencia de MSC

Se utilizan en este ejemplo las referencias de MSC "data_transmission" para designar dos transmisiones de datos satisfactorias que se refieren a la misma definición de MSC.

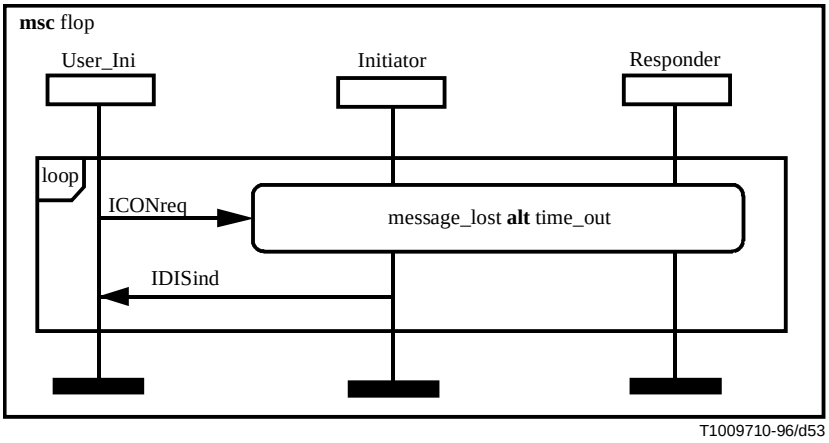



```
msc ref; inst Initiator, Medium, Responder;
        Initiator:          instance;
        Medium:             instance;
        Responder:          instance;
        all:                reference data_transmission;
                            reference data_transmission;
        Initiator:          endinstance;
        Medium:             endinstance;
        Responder:          endinstance;
endmsc;
```

```
msc data_transmission; inst Initiator, Medium, Responder;
        Initiator:      instance;
        Medium:         instance;
        Responder:      instance;
        Initiator:      in IDATreq(d) from env;
                        out MDATreq(DT,data,num) to Medium;
                        set T;
        Medium:         in MDATreq(DT,data,num) from Initiator;
                        out MDATind(DT,data,num) to Responder;
        Responder:      in MDATind(DT,data,num) from Medium;
                        out IDATind(d) to env;
                        out MDATreq(AK,num) to Medium;
        Medium:         in MDATreq(AK,num) from Responder;
                        out MDATind(AK,num) to Initiator;
        Initiator:      in MDATind(AK,num) from Medium;
                        reset T;
        Initiator:      endinstance;
        Medium:         endinstance;
        Responder:      endinstance;
endmsc;
```

6.18 Referencia de MSC con puerta

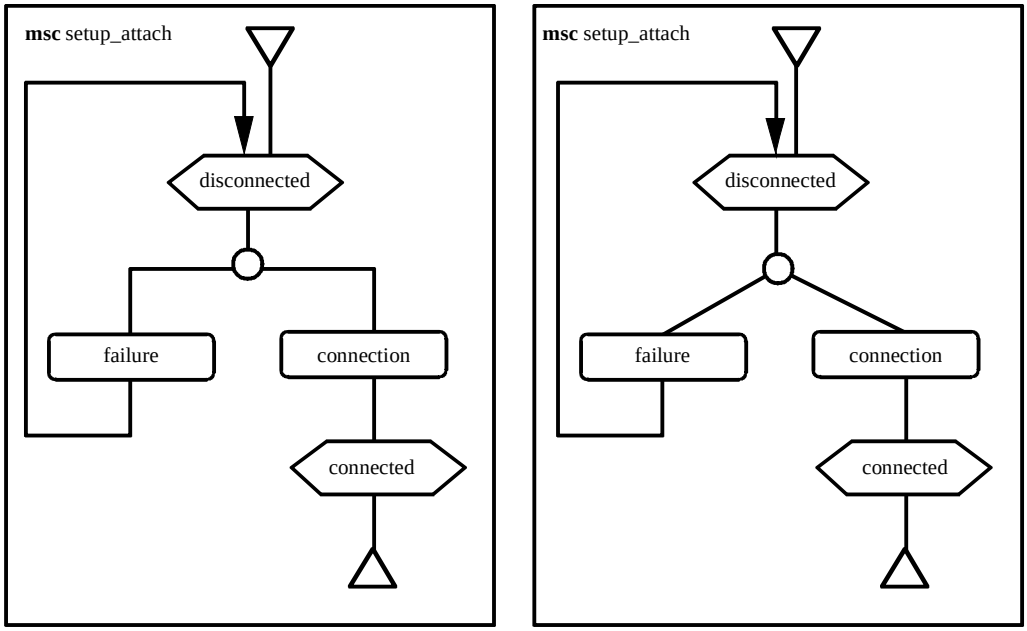
En este ejemplo se representa una referencia de MSC conectada con el exterior a través de una puerta. En el ejemplo 6.21 se facilitan los MSC referenciados "message_lost" y "time_out".




```
msc flop; inst User_Ini, Initiator, Responder;
        User_Ini:       instance;
        Initiator:      instance;
        Responder:      instance;
        all:            loop begin;
            User_Ini:       out ICONreq to reference failed;
            Initiator,
            Responder:      reference failed: message_lost alt time_out;
            Initiator:      out IDISind to User_Ini;
            User_Ini:       in IDISind from Initiator;
                        loop end;
        User_Ini:       endinstance;
        Initiator:      endinstance;
        Responder:      endinstance;
endmsc;
```

6.19 MSC de alto nivel con bucle libre

El MSC "setup_attach" de este ejemplo muestra la modelación del establecimiento de la conexión mediante un bucle libre.

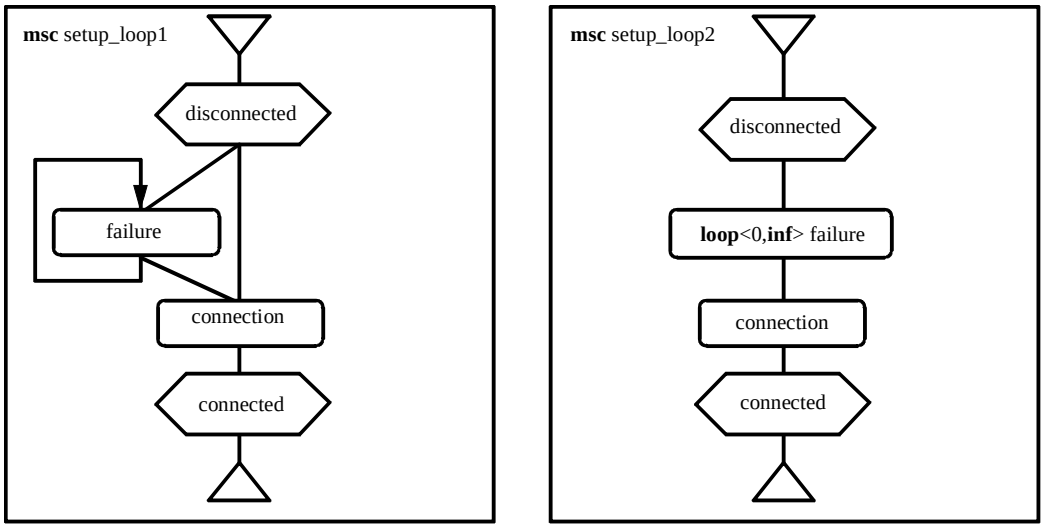


T1009720-96/d54

msc setup_attach;
expr L1;
L1: **condition** disconnected **seq** (L2);
L2: **connect seq** (L3 **alt** L4);
L3: failure **seq** (L1);
L4: connection **seq** (L5);
L5: **condition** connected **seq** (L6);
L6: **end**;
endmsc;

6.20 MSC de alto nivel con bucle

En este ejemplo se muestra la modelación del establecimiento de la conexión mediante un bucle unido a la referencia de MSC "failure".



T1009730-96/d55

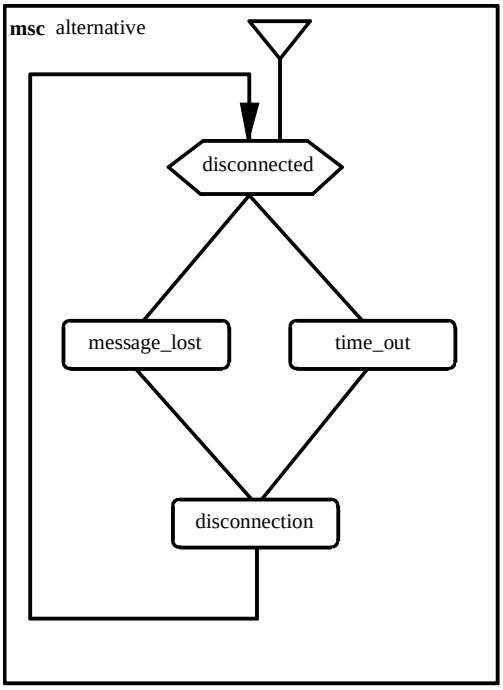
msc setup_loop1;
expr L1;
L1: **condition** disconnected **seq** (L2 **alt** L3);
L2: failure **seq** (L2 **alt** L3);
L3: connection **seq** (L4);

L4: **condition** connected **seq** (L5);
L5: **end**;
endmsc;

msc setup_loop2;
expr L1;
L1: **condition** disconnected **seq** (L2);
L2: (**loop** <0,**inf**> failure) **seq** (L3);
L3: connection **seq** (L4);
L4: **condition** connected **seq** (L5);
L5: **end**;
endmsc;

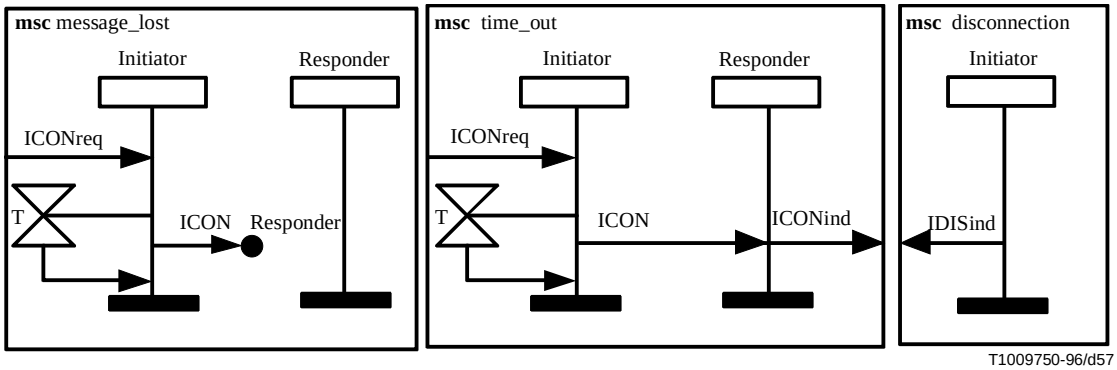
6.21 MSC de alto nivel con composición alternativa

El MSC "alternative" de este ejemplo muestra un constructivo alternativo con paréntesis de corrección en el que las ramas tienen un constructivo de unión correspondiente.



T1009740-96/d56

msc alternative;
expr L1;
L1: **condition** disconnected **seq** (L2 **alt** L3);
L2: message_lost **seq** (L4);
L3: time_out **seq** (L4);
L4: disconnection **seq** (L1);
endmsc;



T1009750-96/d57

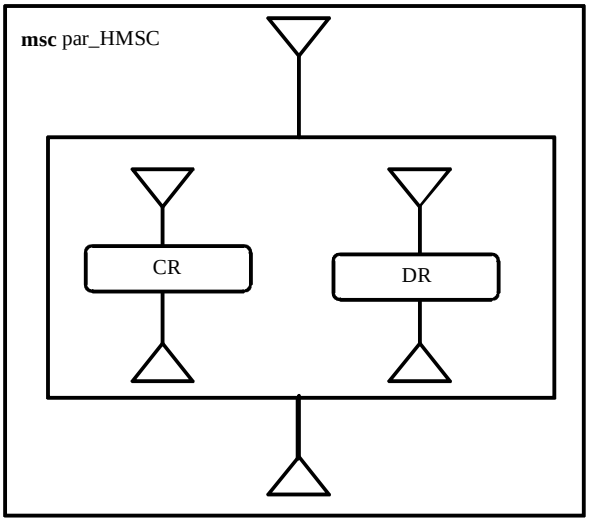
```
msc message_lost; inst Initiator, Responder;
        instance Initiator;
            in ICONreq from env;
            set T;
            out ICON to lost Responder;
            timeout T;
        endinstance;
        instance Responder;
            in ICON Initiator;
        endinstance;
endmsc;

msc time_out; inst Initiator, Responder;
        instance Initiator;
            in ICONreq from env;
            set T;
            out ICON to Responder;
            timeout T;
        endinstance;
        instance Responder;
            in ICON from Initiator;
            out ICONind to env;
        endinstance;
endmsc;

msc disconnection; inst Initiator, Responder;
        instance Initiator;
            out IDISind to env;
        endinstance;
endmsc;
```

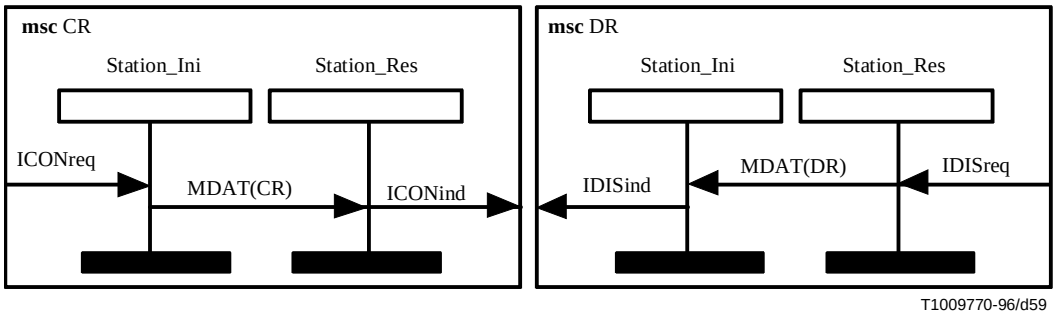
6.22 MSC de alto nivel con composición paralela

En este ejemplo, la petición de conexión procedente del "Initiator" se combina en paralelo con la petición de desconexión procedente del "Responder" utilizando la expresión de HMSC en paralelo.



T1009760-96/d58

```
msc par_HMSC
expr L1;
        L1:   expr L2;
                  L2: CR seq (L3);
                  L3: end;
              endexpr;
        par
              expr L4;
                  L4: DR seq (L5);
                  L5: end;
              endexpr;
        seq (L6);
        L6: end;
endmsc;
```




```
msc CR; inst Station_Ini, Station_Res;
        Station_Ini:    instance;
                        in ICONreq from env;
                        out MDAT(CR) to Station_Res;
                        endinstance;
        Station_Res:    instance;
                        in MDAT(CR) from Station_Ini;
                        out ICONind to env;
                        endinstance;
endmsc;

msc DR; inst Station_Ini, Station_Res;
        Station_Res:    instance;
                        in IDISreq from env;
                        out MDAT(DR) to Station_Ini;
                        endinstance;
        Station_Ini:    instance;
                        in MDAT(DR) from Station_Res;
                        out IDISind to env;
                        endinstance;
endmsc;
```

Anexo A

Índice

Este índice se publica únicamente en la versión inglesa de esta Recomendación.

Reemplazada por una versión más reciente

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**