
Open Modular Interpreter

Casos de uso

Fco. Javier Bohórquez Ogalla

# Índice

# 1. Visión general

En el presente documento se detallan los casos de uso recogidos para el interprete OMI. Para ello los casos de uso son distribuidos en paquetes, y estos modelados mediante un diagrama.

El diagrama de paquetes inicial presenta una organización y distribución general de los paquetes en los que se ha dividido el modelo. Estos paquetes pueden contener diagramas de casos de uso u otros paquetes. Además se establecen relaciones entre estos.

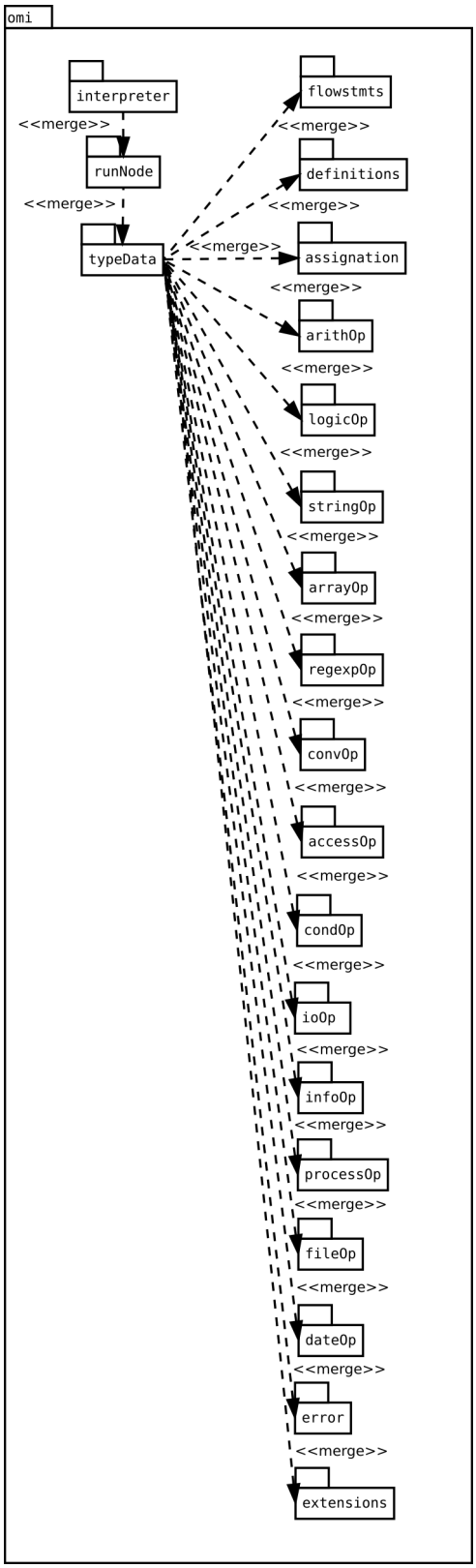
La relación de mezcla (merge) entre dos paquetes significa que el contenido de ambos paquetes será superpuesto, cumplimentando la información que presenta uno con la que ofrece el otro.

Inicialmente se presenta el paquete "interpreter" que recoge los casos de uso que puede llevar a cabo el usuario. Aquí el caso de uso que cabría destacar es el de Interpretar. Este consiste en la interpretación sentencia a sentencia del código fuente facilitado por el usuario. Este caso de uso es extendido según el tipo de sentencia interpretada en cada momento.

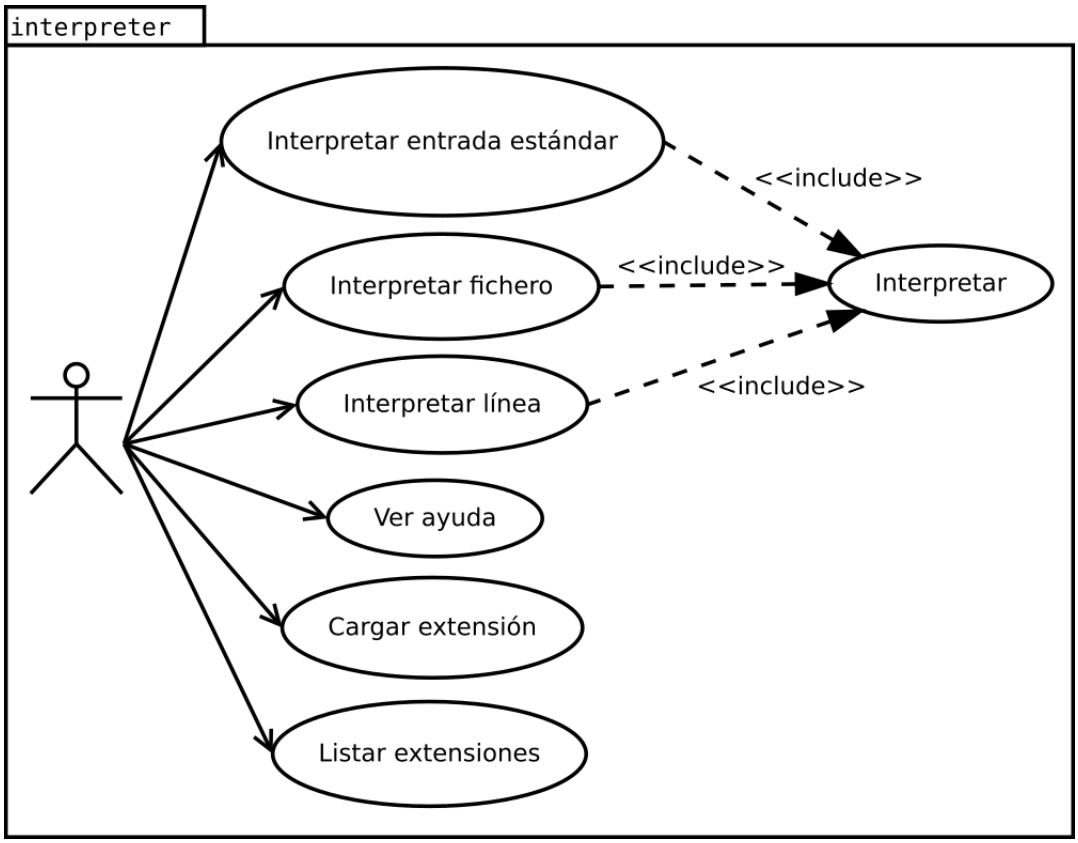
El paquete "interpreter" es completado con el paquete "expresion" que define el caso de uso con el mismo nombre. Una expresión representa una unidad con valor ligado. Los diferentes tipos de sentencias pueden componerse de expresiones que serán valoradas para determinar el comportamiento de la misma. Además una sentencia puede corresponderse con una única expresión. Ante una expresión el sistema debe resolverla estableciendo los operadores de mayor prioridad y dividiéndola así en operaciones de subexpresiones. La valoración de la expresión dependerá del tipo de esta, siendo el caso de uso extendido en este punto.

El paquete "dataType" presenta los tipos de datos como expresiones, y establece cómo se valoran expresiones que codifican los diferentes tipos de datos de forma directa.

Los demás paquetes contienen distintas funcionalidades y características del interprete expuestas de una forma categorizadas. Estos se corresponderán con tipos de sentencias o expresiones.


omi
interpreter
<<merge>>
runNode
<<merge>>
typeData
<<merge>>
flowstmts
<<merge>>
definitions
<<merge>>
assignation
<<merge>>
arithOp
<<merge>>
logicOp
<<merge>>
stringOp
<<merge>>
arrayOp
<<merge>>
regexpOp
<<merge>>
convOp
<<merge>>
accessOp
<<merge>>
condOp
<<merge>>
ioOp
<<merge>>
infoOp
<<merge>>
processOp
<<merge>>
fileOp
<<merge>>
dateOp
<<merge>>
error
<<merge>>
extensions

# 2. Intérprete




## 2.1. Interpretar entrada estándar

**Caso de Uso:** Interpretar entrada estándar.

**Tipo:** General.

**Descripción:** El usuario introduce un bloque de código en forma de cadena de caracteres mediante la entrada estándar. El sistema lo interpreta y ejecuta.

**Actores:** Usuario.

**Precondiciones:** El sistema debe estar esperando un bloque de código mediante la entrada estándar.

**Postcondiciones:** El código introducido es interpretado y ejecutado.

**Escenario principal:**

1. El usuario introduce un bloque de código en la entrada estándar.
2. El sistema obtiene el bloque de código a interpretar mediante la entrada estándar.
3. Incluir (Interpretar).

## 2.2. Interpretar fichero

**Caso de Uso:** Interpretar fichero.

**Tipo:** General.

**Descripción:** El usuario indica un fichero que contiene código y que será interpretado y ejecutado por el sistema.

**Actores:** Usuario.

**Precondiciones:** El sistema espera que se le indique un fichero.

**Postcondiciones:** El fichero es leído y el código en el mismo es interpretado y ejecutado.

**Escenario principal:**

1. El usuario indica la ruta a un fichero.
2. El sistema lee el fichero y obtiene el código en el mismo.
3. Incluir (Interpretar).

**Flujo alternativo:**

2a. El fichero indicado no se encuentra.
    1. El sistema informa del error y finaliza.

## 2.3. Interpretar línea

**Caso de Uso:** Interpretar línea.

**Tipo:** General.

**Descripción:** El usuario introduce bloques de códigos denominados líneas de una forma interactiva. El sistema solicita por la entrada estándar las líneas de código, que serán interpretadas y ejecutadas.

**Actores:** Usuario.

**Precondiciones:** El sistema se encuentra en modo interactivo.

**Postcondiciones:** Se interpreta cada línea introducida por el usuario

**Escenario principal:**

1. El sistema muestra un prompt que indica que espera una línea de código.
2. El usuario introduce una línea de código.
3. El sistema lee de la entrada estándar la línea introducida.
4. Include (Interpretar).

   El sistema repite el caso de uso hasta que se interpreta una sentencia que produzca la salida.

## 2.4. Interpretar

**Caso de Uso:** Interpretar.

**Tipo:** General.

**Nivel:** Subfunción.

**Descripción:** El sistema analiza, interpreta y ejecuta un bloque de código facilitado por el usuario. Para ello comprueba que este cumple con el léxico y la gramática del lenguaje que define, dividiéndolo a su vez en sentencias que serán interpretadas.

**Precondiciones:** El usuario ha introducido un bloque de código.

**Postcondiciones:** El bloque de código es interpretado.

**Puntos de extensión:**

**Sentencia:** Según el tipo de sentencia.

**Escenario principal:**

1. El sistema procesa y comprueba el bloque de código, aplicando la gramática y léxico que define.
2. El sistema obtiene la primera sentencia contenida en el bloque.
3. Sentencia: punto de extensión.

   El sistema repite el paso 3 por cada sentencia en el bloque.

**Flujo alternativo:**

1a. El código no respeta el léxico del lenguaje.
   1. El sistema informa del error y finaliza.

1b. El código no respeta la gramática del lenguaje.
   1. El sistema informa del error y finaliza.

2a. El código no contiene ninguna sentencia.
   1. El sistema finaliza.

## 2.5. Ver ayuda

**Caso de Uso:** Ver ayuda.

**Tipo:** General.

**Descripción:** Se muestra una ayuda que detalla cada opción disponible en el sistema.

**Actores:** Usuario.

**Precondiciones:** Sin precondiciones.

**Postcondiciones:** El sistema muestra un listado que presenta las distintas opciones.

**Escenario principal:**

1. El usuario indica que quiere visualizar la ayuda.
2. El sistema muestra un listado completo de las opciones que presenta.

## 2.6. Cargar extensión

**Caso de Uso:** Cargar extensión.

**Tipo:** General.

**Descripción:** El usuario indica que una extensión que será cargada por el sistema.

**Actores:** Usuario.

**Precondiciones:** Sin precondiciones.

**Postcondiciones:** El sistema cargar la extensión facilitada.

**Escenario principal:**

1. El usuario indica la ruta a la extensión que desea cargar.
2. El sistema carga la extensión para disponer de las distintas opciones que ofrece.

**Flujo alternativo:**

2a. La extensión indicada no se encuentra.
   1. El sistema informa del error.

2b. La extensión indicada no es una extensión válida.
   1. El sistema informa del error.

## 2.7. Listar extensiones

**Caso de Uso:** Listar extensiones.

**Tipo:** General.

**Descripción:** Lista las extensiones que serán cargadas en cada ejecución del sistema.

**Actores:** Usuario.

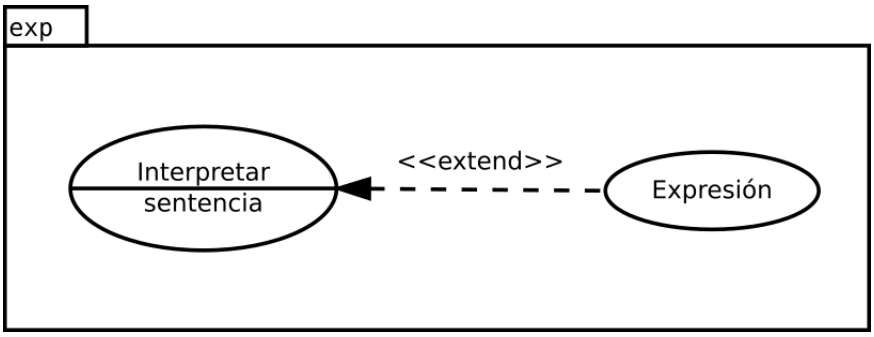
**Precondiciones:** Sin precondiciones.

**Postcondiciones:** El sistema lista las extensiones que serán cargadas.

**Escenario principal:**

1. El usuario indica que desea listar las extensiones cargadas.
2. El sistema lista las extensiones cargadas por defecto.

# 3. Expresiones




**Caso de Uso:** Expresión.

**Tipo:** General.

**Nivel:** Subfunción.

**Descripción:** El sistema valora la expresión según la prioridad de operadores que define.

**Precondiciones:** La expresión cumple con el léxico y gramática del lenguaje.

**Postcondiciones:** El sistema interpreta la expresión obteniendo su valor asociado.

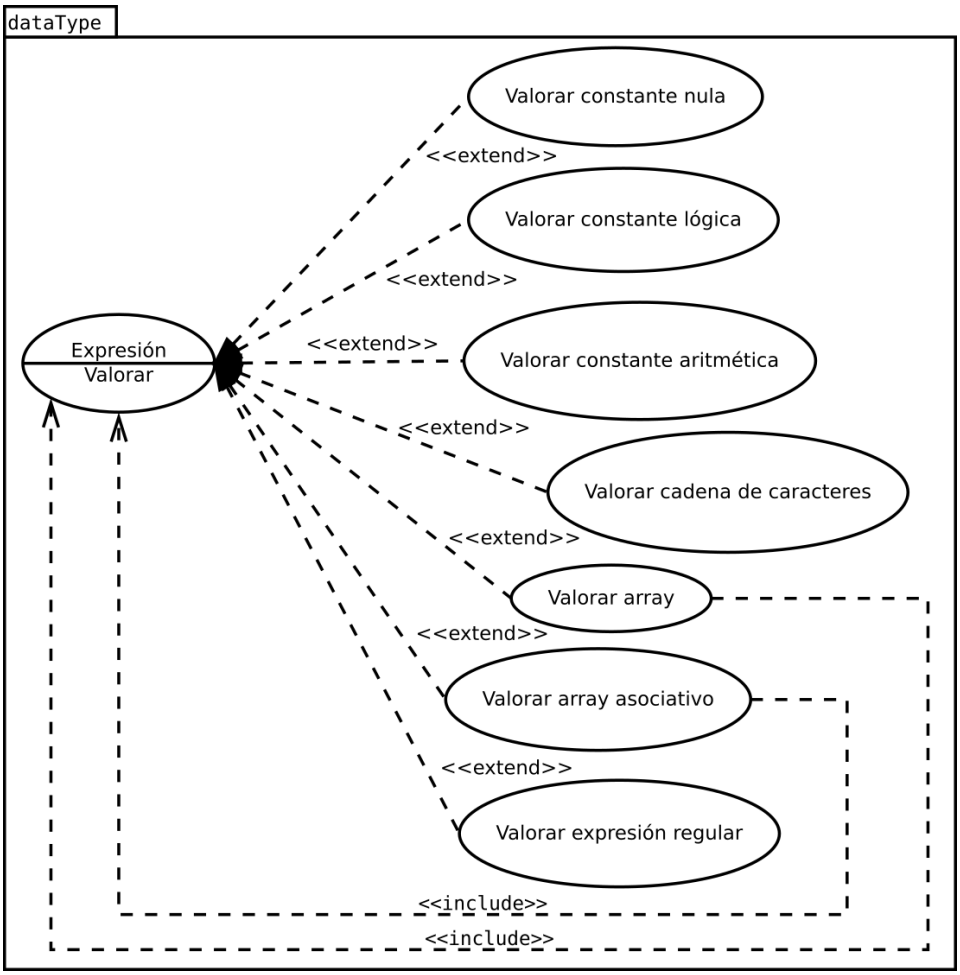
**Puntos de extensión:**

**Valorar:** Según tipo de operador o subexpresión

**Escenario principal:**

1. El sistema determina el operador o subexpresión de mayor prioridad.
2. Valorar: punto de extensión

# 4. Tipos de datos




## 4.1. Valorar constante nula

**Caso de Uso:** Valorar constante nula.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a la constante nula codificada de forma explicita.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es un valor constante nulo.

**Postcondiciones:** El sistema atribuye nulo como valor de la expresión.

**Escenario principal:**

1. El sistema otorga a la expresión en valor nulo.

## 4.2. Valorar constante lógica

**Caso de Uso:** Valorar constante lógica.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a una constante lógica verdadero o falso.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es una constante lógica.

**Postcondiciones:** El sistema atribuye verdadero o falso como valor de la expresión.

**Escenario principal:**

1. El sistema otorga a la expresión el valor lógico verdadero si se corresponde con la expresión “true” o falso si se corresponde con “false”.

## 4.3. Valorar constante aritmética

**Caso de Uso:** Valorar constante aritmética.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión que se corresponde con un número dentro del conjunto de los racionales.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es una constante numérica.

**Postcondiciones:** El sistema atribuye el número como valor de la expresión.

**Escenario principal:**

1. El sistema otorga a la expresión el valor numérico codificado en la misma.

## 4.4. Valorar cadena de caracteres

**Caso de Uso:** Valorar cadena de caracteres.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a una cadena de caracteres constante.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es un valor cadena de caracteres.

**Postcondiciones:** El sistema atribuye la cadena de caracteres como valor de la expresión.

**Escenario principal:**

1. El sistema otorga a la expresión el valor correspondiente a la cadena codificada en la misma.

## 4.5. Valorar array

**Caso de Uso:** Valorar array.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a un array no asociativo.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es un array.

**Postcondiciones:** El sistema atribuye el array como valor de la expresión.

**Escenario principal:**

1. El sistema obtiene la subexpresión correspondiente al primer elemento.
2. Include (Expresión)
3. Atribuye el valor de la subexpresión como último elemento en el array.

   Se repiten los pasos 2-3 secuencialmente para cada elemento.

**Flujo alternativo:**

1a. El array no contiene elementos.
   1. El sistema atribuye como valor de la expresión el array vacío.

## 4.6. Valorar array asociativo

**Caso de Uso:** Array asociativo.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a un array asociativo .

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es un valor array asociativo.

**Postcondiciones:** El sistema atribuye el array asociativo como valor de la expresión.

**Escenario principal:**

1. El sistema obtiene la subexpresión correspondiente al primer elemento.
2. El sistema obtiene la subexpresión correspondiente la clave.
3. Include (Expresión)

4. El sistema obtiene la subexpresión correspondiente al valor.
5. Include (Expresión)
6. Atribuye el par clave/valor como último elemento en el array.

Se repiten los pasos 2-6 secuencialmente para cada elemento.

**Flujo alternativo:**

1a. El array no contiene elementos.
1. El sistema atribuye como valor de la expresión el array vacío.

## 4.7. Valorar expresión regular

**Caso de Uso:** Valor expresión regular.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** El sistema valora una expresión o subexpresión correspondiente a una expresión regular PERL.

**Precondiciones:** El sistema se encuentra valorando una expresión y esta es una expresión regular PERL.

**Postcondiciones:** El sistema atribuye la expresión regular como valor de la expresión PERL.

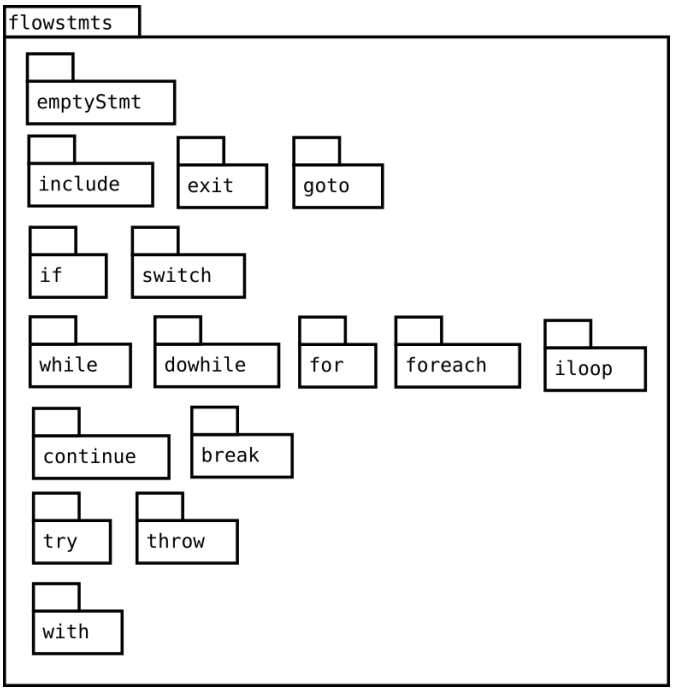
**Escenario principal:**

1. El sistema obtiene la expresión regular codificada.
2. El sistema atribuye la expresión regular como valor de la expresión

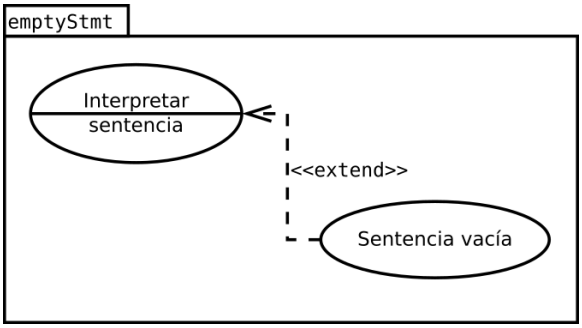
**Flujo alternativo:**

1a. La expresión regular no mantiene una sintaxis PERL.
1. El sistema informa del error y finaliza la interpretación de la sentencia correspondiente.

# 5. Sentencias de control de flujo




## 5.1. Sentencia vacía




**Caso de Uso:** Sentencia vacía.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Se interpreta una sentencia vacía sin producir resultado alguno.

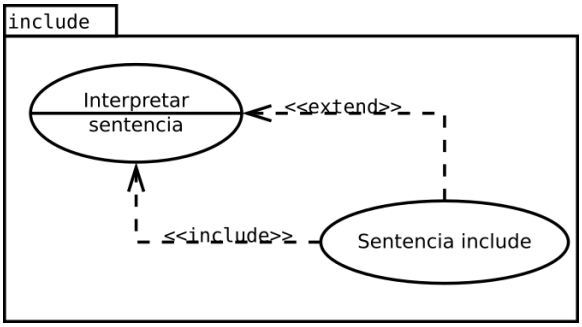
**Precondiciones:** La sentencia interpretada es una sentencia vacía.

**Postcondiciones:** No tiene.

**Escenario principal:**

1. El sistema no realiza ninguna acción.

## 5.2. Sentencia include




**Caso de Uso:** Sentencia include.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Se interpreta una sentencia include, lo que se corresponde con la lectura e interpretación del fichero codificado en la misma.

**Precondiciones:** La sentencia interpretada es una sentencia include.

**Postcondiciones:** El fichero es leído e interpretado

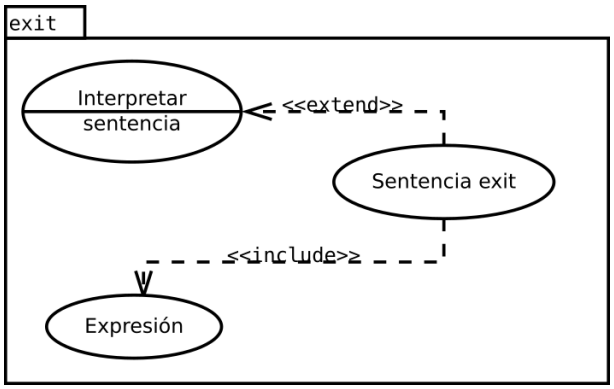
**Escenario principal:**

1. El sistema obtiene el código fuente del fichero.
2. Include (Interpretar).

**Flujo alternativo:**

1a. El fichero no existe.
   1. El sistema informa del error y finaliza la interpretación de la sentencia actual.

## 5.3. Sentencia exit




**Caso de Uso:** Sentencia exit.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Se interpreta una sentencia exit, lo que se corresponde con la salida y finalización del sistema.

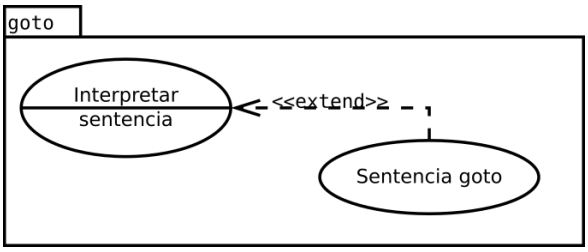
**Precondiciones:** La sentencia interpretada es una sentencia exit.

**Postcondiciones:** Se sale del sistema con el código codificado en la sentencia.

**Escenario principal:**

1. El sistema obtiene la expresión correspondiente al código de salida.
2. Include (Expresión)
3. El sistema finaliza su ejecución devolviendo el valor del código.

## 5.4. Sentencia goto



**Caso de Uso:** Sentencia goto.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Hace que la siguiente sentencia interpretada sea la que esté referenciada por la etiqueta codificada en la sentencia.

**Precondiciones:** La sentencia interpretada es una sentencia goto.

**Postcondiciones:** La siguiente sentencia que se ejecutará se corresponde con la referenciada por la etiqueta indicada.

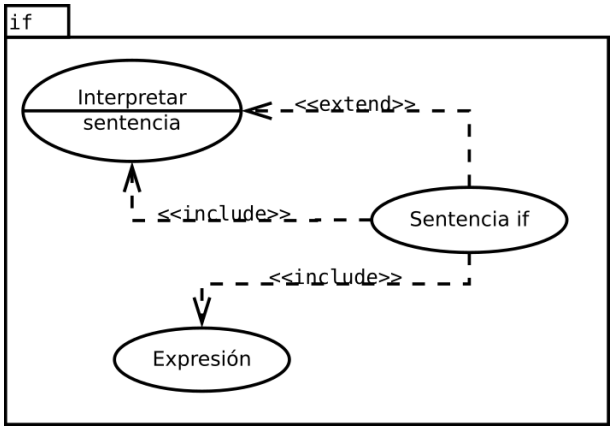
**Escenario principal:**

1. El sistema obtiene la etiqueta codificada en la sentencia.
2. El sistema cambia el flujo de ejecución a la sentencia obtenida.

**Flujo alternativo:**

1a. No existe definida la etiqueta codificada.
   1. El sistema informa del error y finaliza la interpretación de la sentencia actual.

## 5.5. Sentencia if



**Caso de Uso:** Sentencia if.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia if, lo que implica que se interpretará el bloque de sentencias "then" si la expresión de condición es verdadera o el bloque "else" si es falsa.

**Precondiciones:** La sentencia interpretada es una sentencia if.

**Postcondiciones:** La sentencia if queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de condición.
2. Include (Expresión).
3. Si la expresión tiene valor lógico verdadero el sistema obtiene el bloque de sentencias "then".
4. Include (Interpretar).

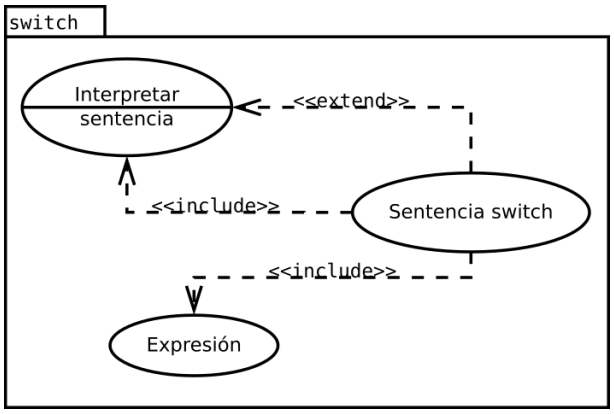
**Flujo alternativo:**

3a. La expresión no tiene valor lógico asociado.
   1. Esta es considerada falsa y se prosigue con el caso de uso.

3b. La expresión tiene valor falso y la sentencia codifica un bloque "else".
   1. El sistema obtiene el bloque de sentencias "else".
   2. Include (Interpretar)

3c. La expresión tiene valor falso y la sentencia no codifica un bloque "else".
   1. El sistema no realiza ninguna acción.

## 5.6. Sentencia switch




**Caso de Uso:** Sentencia switch.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia switch, por lo que se ejecutará el bloque de sentencias correspondiente según el valor de la expresión condición, y todos los bloques siguientes.

**Precondiciones:** La sentencia interpretada es una sentencia switch.

**Postcondiciones:** La sentencia switch queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de condición.
2. Include (Expresión).
3. Se obtiene el bloque de sentencias correspondiente al valor de la expresión.
4. Include (Interpretar).

   Se repite el paso 4 por cada bloque de sentencias de los casos siguientes.

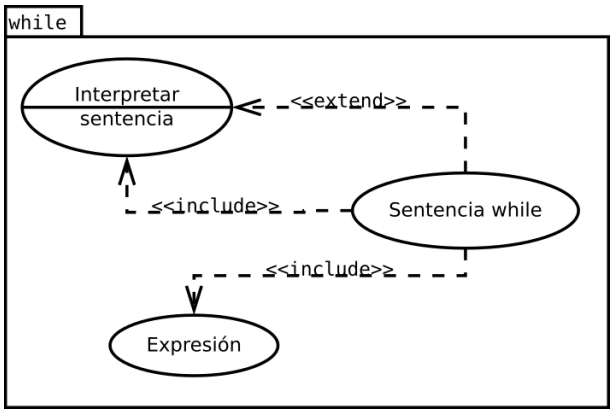
**Flujo alternativo:**

3a. No existe bloque de sentencias para el valor de la expresión, pero existe un bloque de sentencias por defecto.
   1. Se obtiene el bloque de sentencias por defecto.
   2. Include (Interpretar)

3b. No existe bloque de sentencias para el valor de la expresión y tampoco existe un bloque de sentencias por defecto.

1. El sistema no realiza ninguna acción.

## 5.7. Sentencia while




**Caso de Uso:** Sentencia while.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta la sentencia while, lo que implica que se ejecutará el bloque de sentencias mientras el valor de la expresión de condición sea verdadera.

**Precondiciones:** La sentencia interpretada es una sentencia while.

**Postcondiciones:** La sentencia while queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de condición.
2. Include (Expresión).
3. Si la expresión es verdadera obtiene el bloque de sentencias.
4. Include (Interpretar).

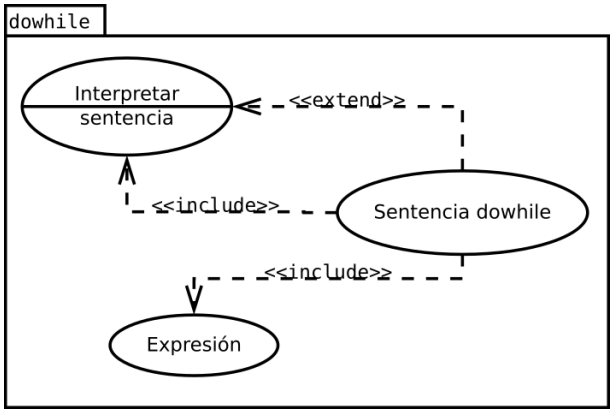
   Se repite el paso 1-4 hasta que la expresión sea valorada como falsa.

**Flujo alternativo:**

3a. La expresión no tiene un valor booleano asociado.

1. Se considera como si tuviera valor falso y se continua con el caso de uso.

## 5.8. Sentencia do...while




**Caso de Uso:** Sentencia do...while.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia do...while, lo que implica que se ejecutará el bloque de sentencias, al menos una vez, y mientras el valor de la expresión de condición sea verdadera.

**Precondiciones:** La sentencia interpretada es una sentencia do...while.

**Postcondiciones:** La sentencia do...while queda interpretada.

**Escenario principal:**

1. El sistema obtiene el bloque de sentencias.
2. Include (Interpretar).
3. El sistema obtiene la expresión de condición.

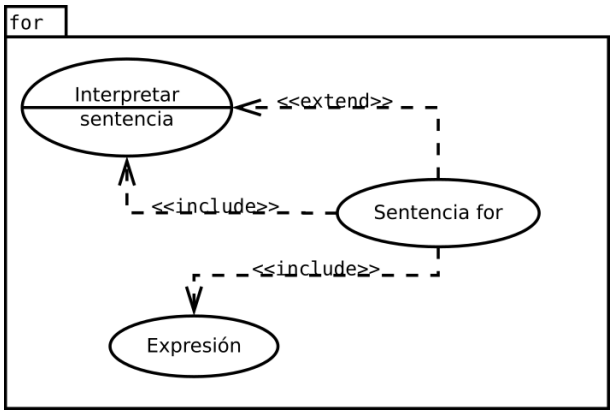
4. Include (Expresión).

Se repite el paso 1-4 hasta que la expresión sea valorada como falsa.

**Flujo alternativo:**

4a. La expresión no tiene un valor booleano asociado.
1. Se considera como si tuviera valor falso y se continua con el caso de uso.

## 5.9. Sentencia for




**Caso de Uso:** Sentencia for.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia for. Para ello se valorará la expresión de inicialización y se ejecutará el bloque de sentencias mientras la expresión de condición tenga un valor verdadero, en cada iteración valorará la expresión de paso.

**Precondiciones:** La sentencia interpretada es una sentencia for.

**Postcondiciones:** La sentencia for queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de inicialización.

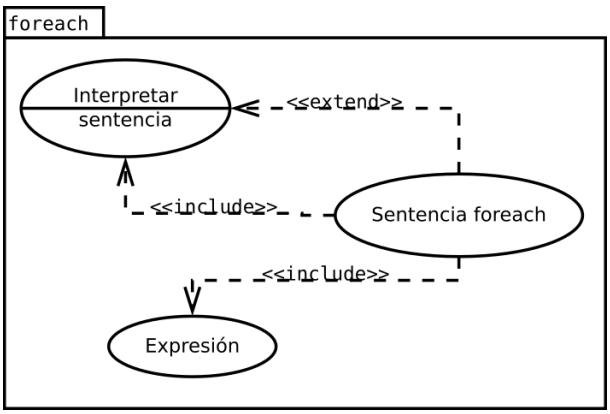
2. Include (Expresión).
3. El sistema obtiene la expresión de condición.
4. Include (Expresión).
5. Si el valor de la expresión condición tiene valor lógico verdadero el sistema obtiene el bloque de sentencias.
6. Include (Interpretar).
7. El sistema obtiene la expresión de paso.
8. Include (Expresión).

   Se repite el paso 3-8 hasta que la expresión de condición sea valorada falsa sea valorada como falsa.

**Flujo alternativo:**

5a. La expresión no tiene un valor booleano asociado.
   1. Se considera como si tuviera valor falso y se continua con el caso de uso.

## 5.10. Sentencia foreach




**Caso de Uso:** Sentencia forearch.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia foreach. Para ello valora la expresión de conjunto y ejecuta el bloque de sentencias por cada uno de los elementos contenidos en el valor devuelto. En cada iteración el elemento será asignado a el identificador codificado en la sentencia foreach.

**Precondiciones:** La sentencia interpretada es una sentencia foreach.

**Postcondiciones:** La sentencia foreach queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de conjunto.
2. Include (Expresión).
3. El sistema obtiene el primer elemento del valor de la expresión.
4. El sistema asigna el valor del elemento al identificador facilitado y obtiene el bloque de sentencias.
5. Include (Interprete)

   Se repite los pasos 3-4 por cada elemento en el valor de la expresión.

**Flujo alternativo:**

3a. La expresión resulta en un conjunto vacío.
   1. El sistema no realiza ninguna acción.

3b. La expresión es una cadena de caracteres.
   1. Se toma como conjunto cada caracter en la cadena.

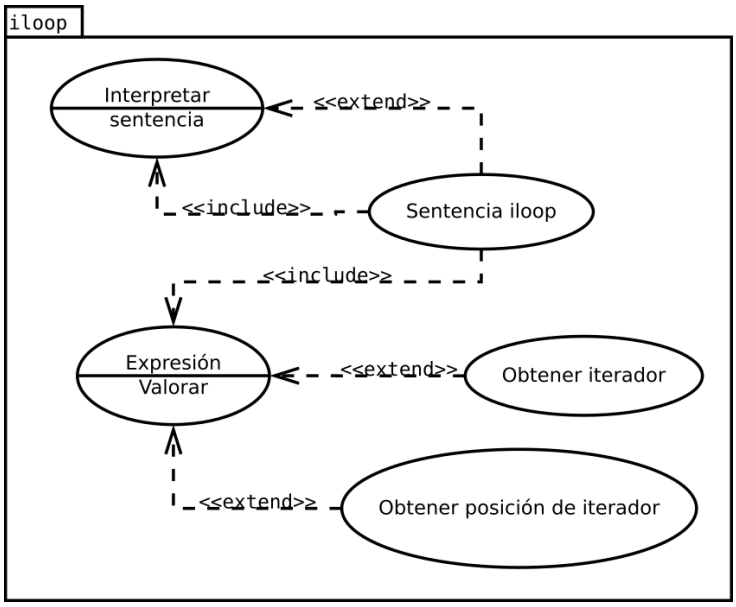
3c. La expresión tiene un valor numérico.
   1. Se toma como conjunto cada número entero desde 0 al valor entero de la expresión.

3d. La expresión tiene un valor lógico.
   1. Si el valor es verdadero lo asigna al identificador y obtiene el bloque de sentencias
   2. Include (Interpretar)

      Repite el caso de uso.

3e. La expresión tiene otro tipo de valor.
   1. El sistema no realiza ninguna acción.

## 5.11. Sentencia de iteración ágil




**Caso de Uso:** Sentencia de iteración ágil.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Interpreta una sentencia de iteración ágil. Para ello valora la expresión de conjunto y ejecuta el bloque de sentencias por cada uno de los elementos contenidos en el valor devuelto. Cada elemento será asignado a una entidad denominada de iterador, también será asignada la posición del elemento en la secuencia a otra entidad denominada posición del iterador.

**Precondiciones:** La sentencia interpretada es una sentencia de iteración ágil.

**Postcondiciones:** La sentencia de iteración ágil queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión de conjunto.
2. Include (Expresión).
3. El sistema obtiene el primer elemento del valor de la expresión.
4. El sistema asigna el valor del elemento al iterador y la posición del mismo a la posición del iterador, luego obtiene el bloque de sentencias para ser interpretadas.

5. Include (Interprete)

   Se repite los pasos 3-4 por cada elemento en el valor de la expresión.

**Flujo alternativo:**

3a. La expresión resulta en un conjunto vacío.
   1. El sistema no realiza ninguna acción.

3b. La expresión es una cadena de caracteres.
   1. Se toma como conjunto cada carácter en la cadena.

3c. La expresión tiene un valor numérico.
   1. Se toma como conjunto cada número entero desde 0 al valor entero de la expresión.

3d. La expresión tiene un valor lógico.
   1. Si el valor es verdadero lo asigna al iterador y se obtiene el bloque de sentencias
   2. Include (Interpretar)

      Repite el caso de uso.

3e. La expresión tiene otro tipo de valor.
   1. El sistema no realiza ninguna acción.

### 5.11.1. Obtener iterador

**Caso de Uso:** Obtener iterador.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** Accede al valor del iterador de una de las sentencias de iteración ágil que se pueden encontrar anidadas. La expresión puede codificar el índice de la sentencia de iteración ágil que se encuentra anidada y sobre la cual se obtendrá el valor del iterador. El índice comienza en 0 y a partir de la última sentencia anidada.

**Precondiciones:** Se valora una expresión de acceso al iterador, además la expresión forma parte de una sentencia que se encuentra dentro del bloque de una sentencia de iteración ágil, o conjunto de estas anidas.

**Postcondiciones:** Se atribuye a la expresión el valor del iterador de la sentencia de iteración ágil indicada.

**Escenario principal:**

1. El sistema obtiene y atribuye a la expresión el valor del iterador de la sentencia de iteración ágil indicada por el índice facilitado.

**Flujo alternativo:**

1a. No se ha facilitado un índice.
   1. Se toma como índice el valor 0.

1b. No se encuentra una sentencia de iteración ágil en el índice dado.
   1. El sistema informa del error y finaliza la interpretación del la sentencia.

### 5.11.2. Obtener posición de iterador

**Caso de Uso:** Obtener posición de iterador.

**Tipo:** Expresión.

**Nivel:** Subfunción.

**Descripción:** Accede al valor de la posición del iterador de una de las sentencias de iteración ágil que se pueden encontrar anidadas. La expresión puede codificar el índice de la sentencia de iteración ágil que se encuentra anidada y sobre la cual se obtendrá el valor de la posición del iterador. El índice comienza en 0 y a partir de la última sentencia anidada.

**Precondiciones:** Se valora una expresión de acceso a la posición del iterador, además la expresión forma parte de una sentencia que se encuentra dentro del bloque de una sentencia de iteración ágil, o conjunto de estas anidas.

**Postcondiciones:** Se atribuye a la expresión el valor de la posición del iterador de la sentencia de iteración ágil indicada.

**Escenario principal:**

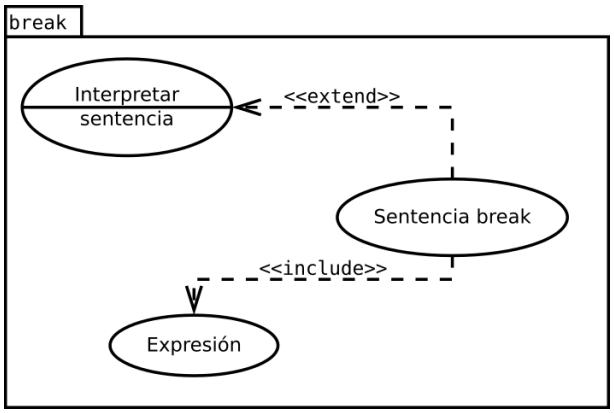
1. El sistema obtiene y atribuye a la expresión el valor de la posición del iterador de la sentencia de iteración ágil indicada por el índice facilitado.

**Flujo alternativo:**

1a. No se ha facilitado un índice.
   1. Se toma como índice el valor 0.

1b. No se encuentra una sentencia de iteración ágil en el índice dado.
   1. El sistema informa del error y finaliza la interpretación del la sentencia.

## 5.12. Sentencia break




**Caso de Uso:** Sentencia break.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Termina la interpretación de un bloque de sentencias o bloques de sentencias anidados. Adicionalmente es posible indicar, codificado en forma de expresión, un índice que determina la posición del bloque de sentencias anidado cuya interpretación finalizará.

**Precondiciones:** La sentencia interpretada es una sentencia break y se encuentra dentro de un bloque de sentencias.

**Postcondiciones:** La sentencia break queda interpretada.

**Escenario principal:**

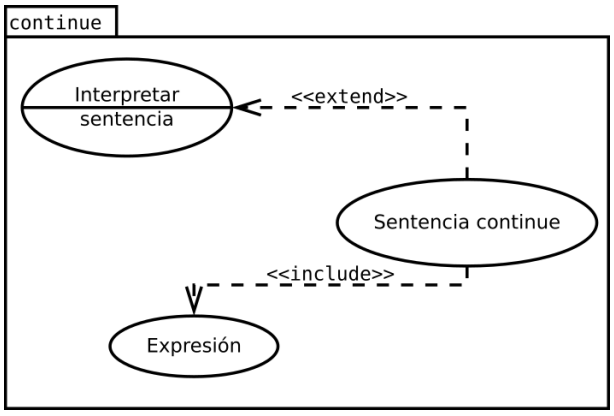
1. El sistema obtiene la expresión correspondiente al índice, que referencia el bloque que finalizará.
2. Include (Expresión).
3. El sistema finaliza la interpretación del bloque de sentencias indicado por el valor de la expresión.
4. El sistema hace que la próxima sentencia que se interprete sea la siguiente al bloque.

**Flujo alternativo:**

1a. No se ha facilitado un índice.
   1. Se toma como índice el valor 0.

3a. No se encuentra un bloque de sentencias en el índice dado.
   1. El sistema informa del error y finaliza la interpretación del la sentencia.

## 5.13. Sentencia continue




**Caso de Uso:** Sentencia continue.

**Tipo:** Sentencia.

**Nivel:** Subfunción.

**Descripción:** Termina la iteración actual de un bloque de sentencias iterativo o bloques de sentencias anidados. Adicionalmente es posible indicar, codificado en forma de expresión, un índice que determina la posición del bloque de sentencias iterativa anidado cuya iteración actual finalizará.

**Precondiciones:** La sentencia interpretada es una sentencia continue y se encuentra dentro de un bloque de sentencias iterativo.

**Postcondiciones:** La sentencia continue queda interpretada.

**Escenario principal:**

1. El sistema obtiene la expresión correspondiente al índice, que referencia el bloque iterativo que finalizará.
2. Include (Expresión).
3. El sistema finaliza la iteración actual del bloque de sentencias indicado por el valor de la expresión.
4. El sistema hace que la próxima sentencia sea la correspondiente al fin de iteración del bloque especificado.

**Flujo alternativo:**

1a. No se ha facilitado un índice.
   1. Se toma como índice el valor 0.

3a. No se encuentra un bloque de sentencias iterativo en el índice dado.
   1. El sistema informa del error y finaliza la interpretación del la sentencia.

## 5.14. Sentencia try...catch [Aún por completar]

Aún por completar.

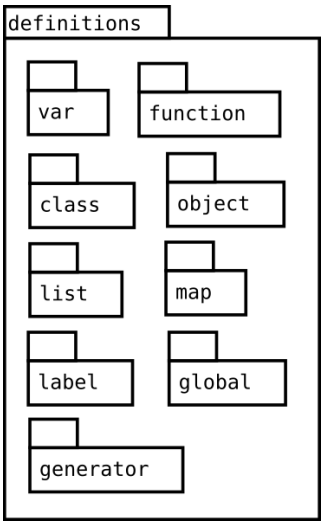
## 5.15. Sentencia throw [Aún por completar]

Aún por completar.

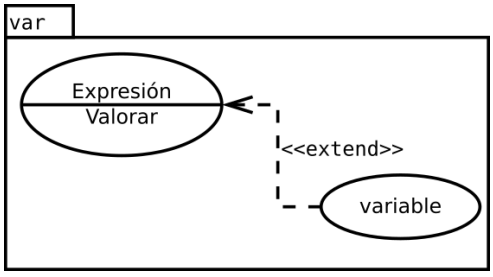
## 5.16. Sentencia with [Aún por completar]

Aún por completar.

# 6. Definiciones




## 6.1. Variable




**Caso de Uso:** Variable.

**Tipo:** Expresiones.

**Nivel:** Subfunción.

**Descripción:** El sistema valora la expresión accediendo al valor guardado en la variable que se corresponde con el identificador codificado.

**Precondiciones:** El sistema se encuentra valorando una expresión.

**Postcondiciones:** A la expresión se le atribuye el valor asociado con el identificador.

**Escenario principal:**

1. El sistema obtiene el valor asociado al identificador codificado y se lo atribuye a la expresión.

**Flujo alternativo:**

1a. El identificador no tiene valor asociado.
    1. Se toma valor nulo.

## 6.2. Funciones [Aún por completar]

Aún por completar.

## 6.3. Clases [Aún por completar]

Aún por completar.

## 6.4. Objetos [Aún por completar]

Aún por completar.

## 6.5. Label [Aún por completar]

Aún por completar.

## 6.6. Definiciones globales [Aún por completar]

Aún por completar.

## 6.7. Generadores [Aún por completar]

Aún por completar.