Open Modular Interpreter

Modelo de comportamiento

Fco. Javier Bohórquez Ogalla

Índice

1. Visión general **4**

2. Diagramas de secuencia del sistema **4**

2.1. Intérprete 4

2.1.1. Interpretar entrada estándar 4

2.1.2. Interpretar línea 5

2.1.3. Interpretar fichero 5

2.1.4. Ver ayuda 6

2.1.5. Cargar extensión 6

2.1.6. Listar extensiones 7

2.1.7. Iniciar interpretación red 7

2.1.8. Obtener pasos interpretación red 8

2.2. runTree 8

2.2.1. Enviar código fuente 8

2.2.2. Siguiente paso 9

2.2.3. Siguiente sentencia 9

2.2.4. Activar ejecución automática 10

2.2.5. Desctivar ejecución automática 10

2.2.6. Limpiar salida 11

2.2.7. Ver infomación de nodo 11

2.2.8. Ver contenido tabla de símbolo 12

2.2.9. Guardar código fuente 12

2.2.10. Abrir código fuente 13

3. Contratos de operaciones del sistema 13

3.1. Intérprete 13

3.1.1. Operación inicio_interprete_stdin 13

3.1.2. Operación inicio_interprete_linea 15

3.1.3. Operación interpretar_cadena 16

3.1.4. Operación interpretar_fichero 17

3.1.5. Operación iniciar_interpretacion_red 19

3.1.6. Operación obtener_paso_interpretacion_red 21

3.2. runTree 22

3.2.1. Operación enviar_codigo_fuente 22

3.2.2. Operación siguiente_paso 23

3.2.3. Operación siguiente_sentencia 24

1. Visión general

El presente documento constituye el modelo de comportamiento del sistema.

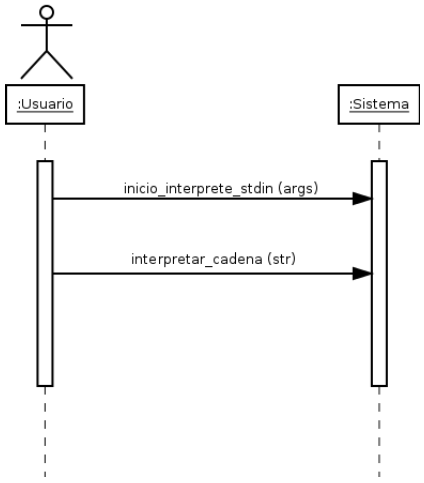
Partiendo de los casos de usos se presentan los diagramas de secuencias que modelan los eventos que el sistema puede recibir del usuario y los valores de retorno que produce como consecuencia de estos. Esta sección considera tanto al sistema intérprete como el cliente.

A partir de los diagramas de secuencia del sistema se obtienen las operaciones que este presenta. Luego se describen los contratos de cada una de las operaciones.

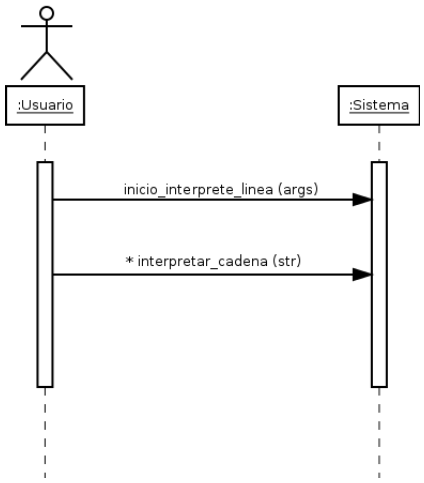
2. Diagramas de secuencia del sistema

2.1. Intérprete

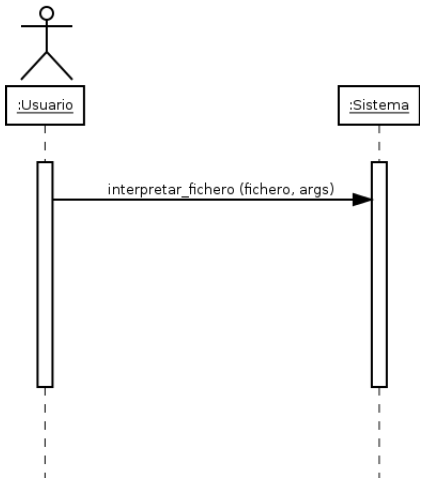
2.1.1. Interpretar entrada estándar



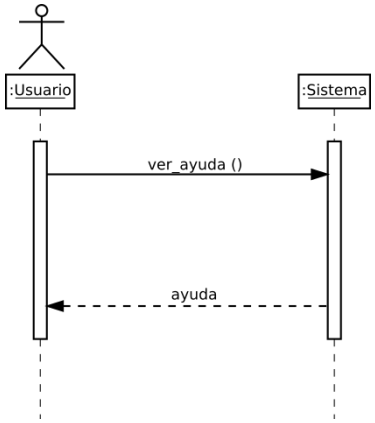
2.1.2. Interpretar línea



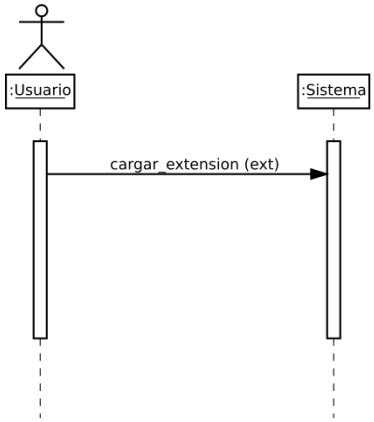
2.1.3. Interpretar fichero



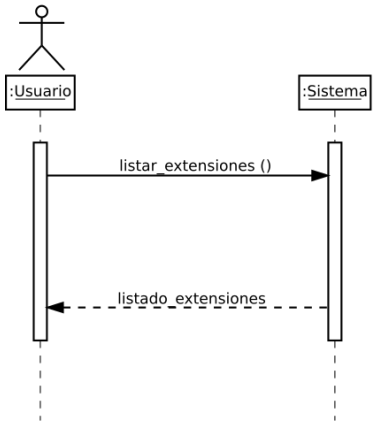
2.1.4. Ver ayuda



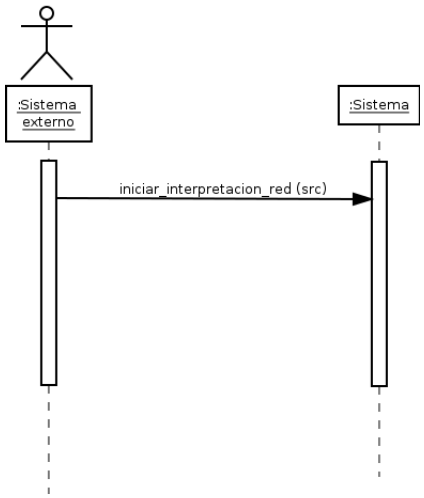
2.1.5. Cargar extensión



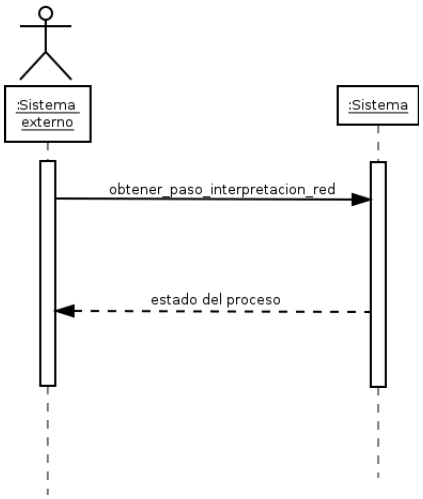
2.1.6. Listar extensiones



2.1.7. Iniciar interpretación red

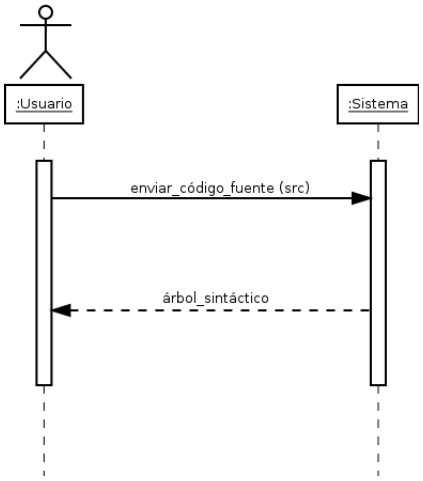


2.1.8. Obtener pasos interpretación red

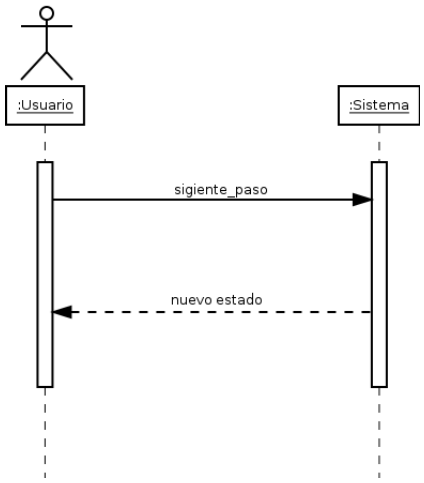


2.2. runTree

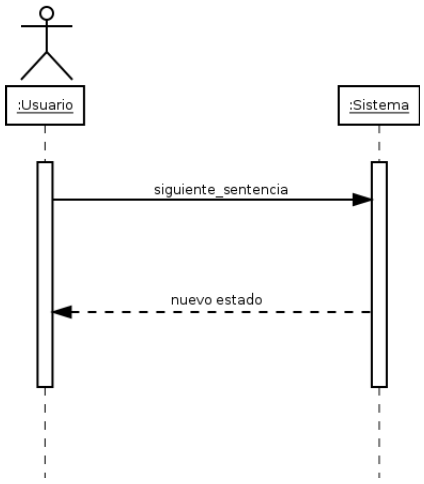
2.2.1. Enviar código fuente



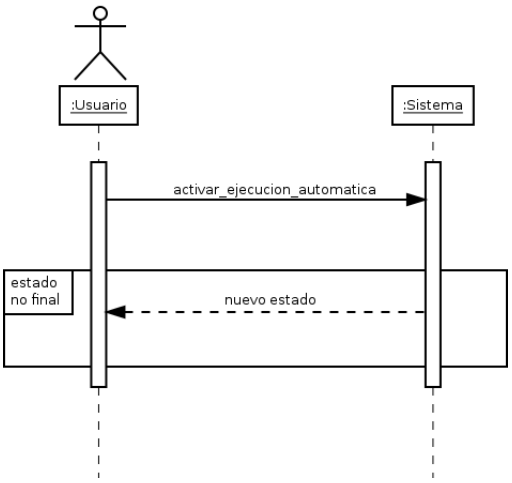
2.2.2. Siguiente paso



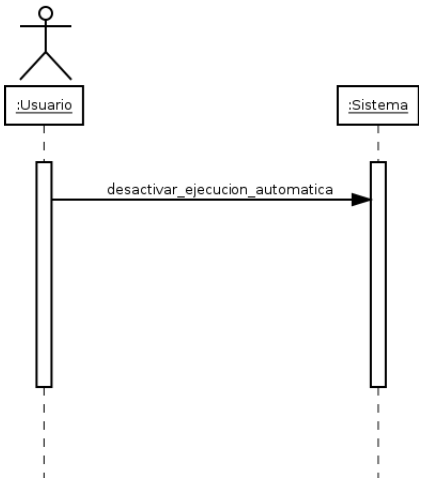
2.2.3. Siguiente sentencia



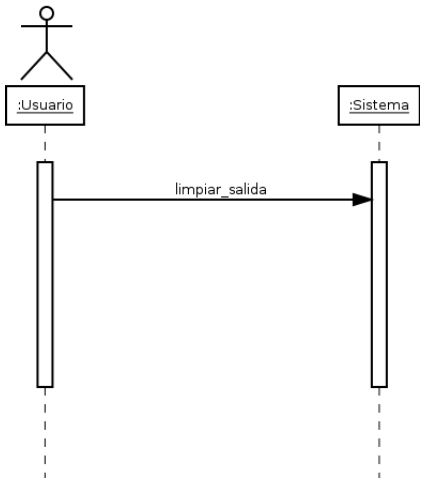
2.2.4. Activar ejecución automática



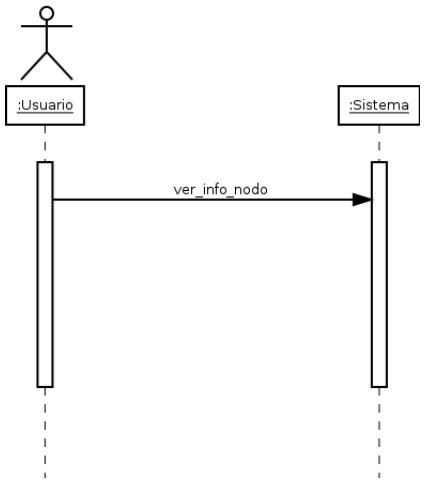
2.2.5. Desctivar ejecución automática



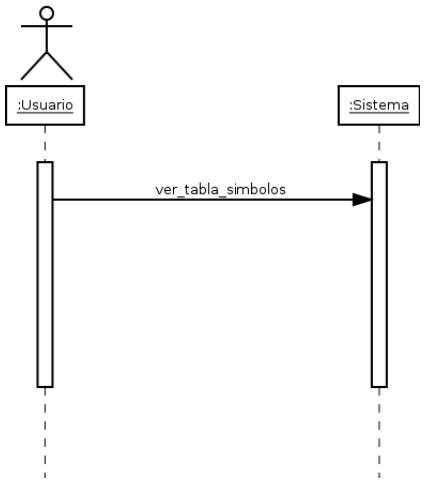
2.2.6. Limpiar salida



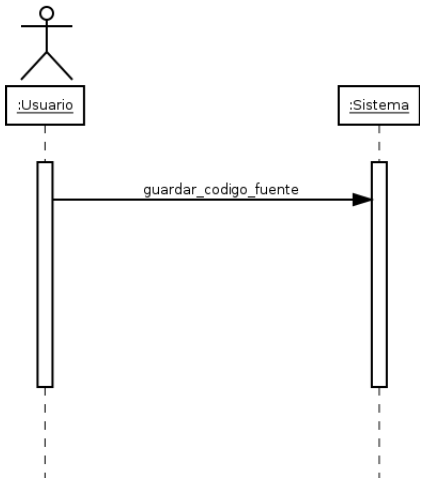
2.2.7. Ver infomación de nodo



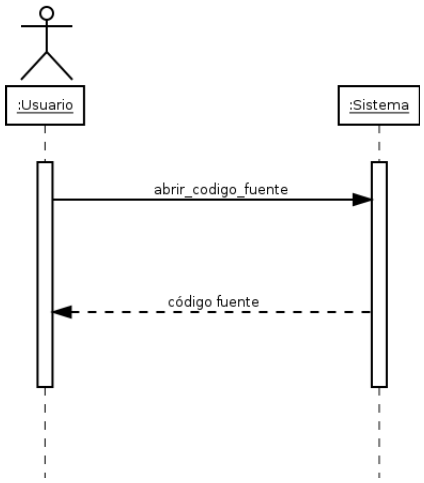
2.2.8. Ver contenido tabla de símbolo



2.2.9. Guardar código fuente



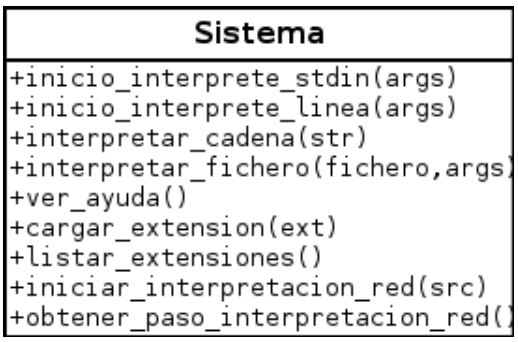
2.2.10. Abrir código fuente



3. Contratos de operaciones del sistema

En esta sección se listan las operaciones del sistema y se detallan los contratos de aquellas que implican un cambio en la estructura de datos interna del programa.

3.1. Intérprete



3.1.1. Operación inicio_interprete_stdin

Nombre: inicio_interprete_stdin(args)

Responsabilidades: Iniciar la interpretación de la entrada estándar.

Referencias Cruzadas: Caso de Uso: Interpretar entrada estándar

Precondiciones: No tiene.

Postcondiciones:

- Se creó una instancia "i" de "interpreter" (creación de objeto).
- Se inicializaron los atributos de "i".
- Se crearon las instancias "$a_0...a_n$" de "arg" por cada argumento en "args" (creación de objeto).
- Se asignó "args_i" a "$a_i.value$" ($a_i.value = \text{args}_i$) (modificación de atributos).
- Se asoció los "arg" "$a_0...a_n$" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "p" de "parser" (creación de objeto).
- Se inicializaron los atributos de "p".
- Se creó una instancia "s" de "scanner" (creación de objeto).
- Se inicializaron los atributos de "s".
- Se asoció el "scanner" "s" al objeto "p" de "parser" (creación de enlace).
- Se asoció el "parser" "p" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "c" de "context" (creación de objeto).
- Se inicializaron los atributos de "c".
- Se creó una instancia "var" de "varSymbols" (creación de objeto).
- Se inicializaron los atributos de "var".
- Se asoció el "varSymbols" "var" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "func" de "funcSymbols" (creación de objeto).
- Se inicializaron los atributos de "func".
- Se asoció el "funcSymbols" "func" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "class" de "classSymbols" (creación de objeto).
- Se inicializaron los atributos de "class".
- Se asoció el "classSymbols" "class" al objeto "c" de "context" (creación de enlace).
- Se asoció el "context" "c" al objeto "i" de "interpreter" (creación de enlace).

3.1.2. Operación inicio_interprete_linea

Nombre: inicio_interprete_linea(args)

Responsabilidades: Iniciar la interpretación interactiva línea a línea.

Referencias Cruzadas: Caso de Uso: Interpretar línea

Precondiciones: No tiene.

Postcondiciones:

- Se creó una instancia "i" de "interpreter" (creación de objeto).
- Se inicializaron los atributos de "i".
- Se crearon las instancias "$a_0...a_n$" de "arg" por cada argumento en "args" (creación de objeto).
- Se asignó "args_i" a "$a_i.value$" ($a_i.value$ =args_i) (modificación de atributos).
- Se asoció los "arg" "$a_0...a_n$" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "p" de "parser" (creación de objeto).
- Se inicializaron los atributos de "p".
- Se creó una instancia "s" de "scanner" (creación de objeto).
- Se inicializaron los atributos de "s".
- Se asoció el "scanner" "s" al objeto "p" de "parser" (creación de enlace).
- Se asoció el "parser" "p" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "c" de "context" (creación de objeto).
- Se inicializaron los atributos de "c".
- Se creó una instancia "var" de "varSymbols" (creación de objeto).
- Se inicializaron los atributos de "var".
- Se asoció el "varSymbols" "var" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "func" de "funcSymbols" (creación de objeto).
- Se inicializaron los atributos de "func".
- Se asoció el "funcSymbols" "func" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "class" de "classSymbols" (creación de objeto).

- Se inicializaron los atributos de “class”.
- Se asoció el “classSymbols” “class” al objeto “c” de “context” (creación de enlace).
- Se asoció el “context” “c” al objeto “i” de “interpreter” (creación de enlace).

3.1.3. Operación interpretar_cadena

Nombre: interpretar_cadena(str)

Responsabilidades: Interpreta el contenido fuente almacenado en la cadena “str”

Referencias Cruzadas:

- Caso de Uso: Interpretar entrada estándar
- Caso de Uso: Interpretar línea

Precondiciones:

- Se creó un “interpreter” “i”.
- Se creó y asoció una instancia de “parser” y “scanner” a “i”.
- Se creó y asoció una instancia de “varSymblos”, “funcSymbols” y “classSymbols” a “i”.

Postcondiciones:

- Se creó una instancia “s” de “source” (creación de objeto).
- Se asignó “str” a “s.src” (s.src = str) (modificación de atributos)
- Se asoció “s” al “scanner” componente del “interpreter” “i” (creación de enlace).
- Se creó un conjunto “$t_0...t_n$” de “token” a partir del análisis léxico (creación de objeto).
- Se creó un conjunto “$n_0...n_n$” de “runNode” a partir del análisis sintáctico (creación de objetos).
- Se asoció “$n_i \epsilon n_0...n_n$” a “$n_k \epsilon n_0...n_n$” para construir el árbol sintáctico (creación de enlace).
- Se asoció “$n_r \epsilon n_0...n_n$”, raíz del árbol sintáctico, al “interprter” “i” (creación de enlace).

- Se creó un conjunto "$var_0...var_n$" de "refNode" correspondientes a las variables definidas en el contenido fuente (creación de objetos).
- Se creó un conjunto "$val_0...val_n$" de "runNode" correspondientes a los valores asignados a las variables definidas en el contenido fuente (creación de objetos).
- Se asoció "$val_i \epsilon\ val_0...val_n$" a "$var_i \epsilon\ var_0...var_n$" donde val_i es el valor de la variable var_i (creación de enlace).
- Se asoció todo "refNode" "$var_0...var_n$" al componente "varSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$func_0...func_n$" de "refNode" correspondientes a las funciones con identificador definidas en el contenido fuente (creación de objetos).
- Se asoció "$func_i \epsilon\ func_0...func_n$" a "$n_i \epsilon\ n_0...n_n$" donde n_i es un "funcNode" correspondiente a la definición de la función $func_i$ (creación de enlaces).
- Se asoció todo "refNode" "$func_0...func_n$" al componente "funcSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$class_0...class_n$" de "refNode" correspondientes a las clases definidas en el contenido fuente (creación de objetos).
- Se asoció "$class_i \epsilon\ class_0...class_n$" a "$n_i \epsilon\ n_0...n_n$" donde n_i es un "classNode" correspondiente a la definición de la clase $class_i$ (creación de enlaces).
- Se asoció todo "refNode" "$class_0...class_n$" al componente "classSymbols" de "i" (creación de enlace).

3.1.4. Operación interpretar_fichero

Nombre: interpretar_fichero(fichero, args)

Responsabilidades: Interpreta el contenido fuente almacenado en "fichero"

Referencias Cruzadas: Caso de Uso: Interpretar fichero

Precondiciones: No tiene.

Postcondiciones:

- Se creó una instancia "i" de "interpreter" (creación de objeto).

- Se inicializaron los atributos de “i”.
- Se creó el conjunto “$a_0...a_n$” de “arg” según el número de argumentos en “args” (creación de objeto).
- Se asignó “args_i” a “$a_i.value$” ($a_i.value$ =args_i) (modificación de atributos).
- Se asoció los “arg” “$a_0...a_n$” al objeto “i” de “interpreter” (creación de enlace).
- Se creó una instancia “p” de “parser” (creación de objeto).
- Se inicializaron los atributos de “p”.
- Se creó una instancia “s” de “scanner” (creación de objeto).
- Se inicializaron los atributos de “s”.
- Se asoció el “scanner” “s” al objeto “p” de “parser” (creación de enlace).
- Se asoció el “parser” “p” al objeto “i” de “interpreter” (creación de enlace).
- Se creó una instancia “c” de “context” (creación de objeto).
- Se inicializaron los atributos de “c”.
- Se creó una instancia “var” de “varSymbols” (creación de objeto).
- Se inicializaron los atributos de “var”.
- Se asoció el “varSymbols” “var” al objeto “c” de “context” (creación de enlace).
- Se creó una instancia “func” de “funcSymbols” (creación de objeto).
- Se inicializaron los atributos de “func”.
- Se asoció el “funcSymbols” “func” al objeto “c” de “context” (creación de enlace).
- Se creó una instancia “class” de “classSymbols” (creación de objeto).
- Se inicializaron los atributos de “class”.
- Se asoció el “classSymbols” “class” al objeto “c” de “context” (creación de enlace).
- Se asoció el “context” “c” al objeto “i” de “interpreter” (creación de enlace).
- Se creó una instancia “src” de “source” (creación de objeto).
- Se asignó el contenido de “fichero” a “src.src” (src.src = fichero) (modificación de atributos)
- Se asoció “src” al “scanner” “s” componente del “interpreter” “i” (creación de enlace).
- Se creó un conjunto “$t_0...t_n$” de “token” a partir del análisis léxico (creación de objeto).

- Se creó un conjunto "$n_0...n_n$" de "runNode" a partir del análisis sintáctico (creación de objetos).
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico (creación de enlace).
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "interprter" "i" (creación de enlace).
- Se creó un conjunto "$var_0...var_n$" de "refNode" correspondientes a las variables definidas en el contenido fuente (creación de objetos).
- Se creó un conjunto "$val_0...val_n$" de "runNode" correspondientes a los valores asignados a las variables definidas en el contenido fuente (creación de objetos).
- Se asoció "$val_i \epsilon val_0...val_n$" a "$var_i \epsilon var_0...var_n$" donde val_i es el valor de la variable var_i (creación de enlace).
- Se asoció todo "refNode" "$var_0...var_n$" al componente "varSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$func_0...func_n$" de "refNode" correspondientes a las funciones con identificador definidas en el contenido fuente (creación de objetos).
- Se asoció "$func_i \epsilon func_0...func_n$" a "$n_i \epsilon n_0...n_n$" donde n_i es un "funcNode" correspondiente a la definición de la función $func_i$ (creación de enlaces).
- Se asoció todo "refNode" "$func_0...func_n$" al componente "funcSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$class_0...class_n$" de "refNode" correspondientes a las clases definidas en el contenido fuente (creación de objetos).
- Se asoció "$class_i \epsilon class_0...class_n$" a "$n_i \epsilon n_0...n_n$" donde n_i es un "classNode" correspondiente a la definición de la clase $class_i$ (creación de enlaces).
- Se asoció todo "refNode" "$class_0...class_n$" al componente "classSymbols" de "i" (creación de enlace).

3.1.5. Operación iniciar_interpretacion_red

Nombre: iniciar_interpretacion_red (src)

Responsabilidades: Iniciar la interpretación de una petición por red.

Referencias Cruzadas: Caso de Uso: Iniciar interpretación red

Precondiciones: No tiene.

Postcondiciones:

- Se creó una instancia "i" de "interpreter" (creación de objeto).
- Se inicializaron los atributos de "i".
- Se creó una instancia "p" de "parser" (creación de objeto).
- Se inicializaron los atributos de "p".
- Se creó una instancia "s" de "scanner" (creación de objeto).
- Se inicializaron los atributos de "s".
- Se asoció el "scanner" "s" al objeto "p" de "parser" (creación de enlace).
- Se asoció el "parser" "p" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "c" de "context" (creación de objeto).
- Se inicializaron los atributos de "c".
- Se creó una instancia "var" de "varSymbols" (creación de objeto).
- Se inicializaron los atributos de "var".
- Se asoció el "varSymbols" "var" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "func" de "funcSymbols" (creación de objeto).
- Se inicializaron los atributos de "func".
- Se asoció el "funcSymbols" "func" al objeto "c" de "context" (creación de enlace).
- Se creó una instancia "class" de "classSymbols" (creación de objeto).
- Se inicializaron los atributos de "class".
- Se asoció el "classSymbols" "class" al objeto "c" de "context" (creación de enlace).
- Se asoció el "context" "c" al objeto "i" de "interpreter" (creación de enlace).
- Se creó una instancia "src" de "source" (creación de objeto).
- Se asignó el contenido de "fichero" a "src.src" (src.src = fichero) (modificación de atributos)
- Se asoció "src" al "scanner" "s" componente del "interpreter" "i" (creación de enlace).
- Se creó un conjunto "$t_0...t_n$" de "token" a partir del análisis léxico (creación de objeto).
- Se creó un conjunto "$n_0...n_n$" de "runNode" a partir del análisis sintáctico (creación de objetos).

- Se asoció "$n_i \in n_0...n_n$" a "$n_k \in n_0...n_n$" para construir el árbol sintáctico (creación de enlace).
- Se asoció "$n_r \in n_0...n_n$", raíz del árbol sintáctico, al "interprter" "i" (creación de enlace).

3.1.6. Operación obtener_paso_interpretacion_red

Nombre: obtener_paso_interpretacion_red

Responsabilidades: Obtiene el siguiente paso de la interpretación de una petición por red abierta.

Referencias Cruzadas: Caso de Uso: Obtener pasos interpretación red

Precondiciones:

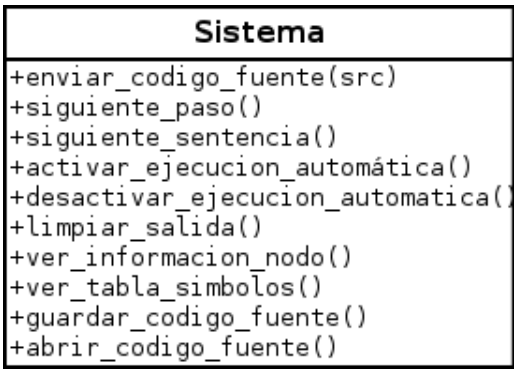
- Se creó un "interpreter" "i".
- Se creó y asoció una instancia de "parser" y "scanner" a "i".
- Se creó y asoció una instancia de "varSymblos", "funcSymbols" y "classSymbols" a "i".
- Se creó una instancia "src" de "source".
- Se asoció "src" al "scanner" "s" componente del "interpreter" "i".
- Se creó un conjunto "$t_0...t_n$" de "token" a partir del análisis léxico.
- Se creó un conjunto "$n_0...n_n$" de "runNode" a partir del análisis sintáctico.
- Se asoció "$n_i \in n_0...n_n$" a "$n_k \in n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \in n_0...n_n$", raíz del árbol sintáctico, al "interprter" "i".

Postcondiciones:

- Se creó un conjunto "$var_0...var_n$" de "refNode" correspondientes a las variables definidas en el paso correspondiente (creación de objetos).
- Se creó un conjunto "$val_0...val_n$" de "runNode" correspondientes a los valores asignados a las variables definidas en el paso correspondiente (creación de objetos).
- Se asoció "$val_i \in val_0...val_n$" a "$var_i \in var_0...var_n$" donde val_i es el valor de la variable var_i (creación de enlace).

- Se asoció todo "refNode" "$var_0...var_n$" al componente "varSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$func_0...func_n$" de "refNode" correspondientes a las funciones con identificador definidas en el paso correspondiente (creación de objetos).
- Se asoció "$func_i \in func_0...func_n$" a "$n_i \in n_0...n_n$" donde n_i es un "funcNode" correspondiente a la definición de la función $func_i$ (creación de enlaces).
- Se asoció todo "refNode" "$func_0...func_n$" al componente "funcSymbols" de "i" (creación de enlace).
- Se creó un conjunto "$class_0...class_n$" de "refNode" correspondientes a las clases definidas en el paso correspondiente (creación de objetos).
- Se asoció "$class_i \in class_0...class_n$" a "$n_i \in n_0...n_n$" donde n_i es un "classNode" correspondiente a la definición de la clase $class_i$ (creación de enlaces).
- Se asoció todo "refNode" "$class_0...class_n$" al componente "classSymbols" de "i" (creación de enlace).

3.2. runTree



3.2.1. Operación enviar_codigo_fuente

Nombre: enviar_codigo_fuente (src)

Responsabilidades: Envía código fuente para una interpretación por red.

Referencias Cruzadas: Caso de Uso: Enviar código fuente

Precondiciones: No tiene.

Postcondiciones:

- Se creó un "tree" "t" a partir del código enviado a interpretación (creación de objeto).
- Se creó un conjunto "$n_0...n_n$" de "node" a partir de los datos recibidos de la petición (creación de objeto).
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico (creación de enlace).
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t" (creación de enlace).
- Se inicializó el atributo "canvas" del "tree" 't" para dibujar el árbol correspondiente a los nodos " $n_0...n_n$".
- Se creó un "symbols" "s" (creación de objeto).
- Se creó tres instancias de "table": "vars", "funcs" y "class" (creación de objeto).
- Se asociarón "vars", "funcs" y "class" a "s" (creación de enlace).

3.2.2. Operación siguiente_paso

Nombre: siguiente_paso

Responsabilidades: Obtiene el siguiente paso del proceso de interpretación abierto.

Referencias Cruzadas: Caso de Uso: Siguiente paso

Precondiciones:

- Se creó un "tree" "t".
- Se creó un conjunto "$n_0...n_n$" de "node".
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t".
- Se creó un "symbols" "s".
- Se creó tres instancias de "table": "vars", "funcs" y "class".

- Se asociarón “vars”, “funcs” y “class” a “s”.

Postcondiciones:

- Se creó el conjunto “$r_0...r_n$” de ‘refs” según el nuevo paso del proceso de interpretación (creación de objeto).
- Se creó el conjunto “$v_0...v_m$” de ‘node” correspondiente a los valores generados en el paso del proceso de interpretación (creación de objeto).
- Se asoció “v_i” a “r_i” como valor de la refercia (creación de enlace).
- Se asoció “$r_i \epsilon r_0...r_n$” a “vars”, “funcs” o “class” según el paso del proceso de interpretación (creación de enlace).
- Se actualizó el valor del atributo “canvas” de “s” para reflejar el nuevo estado tras la ejecución del paso.
- Se actualizó el valor del atributo “canvas” de “t” para reflejar el nuevo estado tras la ejecución del paso.

3.2.3. Operación siguiente_sentencia

Nombre: siguiente_sentencia

Responsabilidades: Obtiene la siguiente sentencia dentro del proceso de interpretación abierto.

Referencias Cruzadas: Caso de Uso: Siguiente sentencia

Precondiciones:

- Se creó un “tree” “t”.
- Se creó un conjunto “$n_0...n_n$” de “node”.
- Se asoció “$n_i \epsilon n_0...n_n$” a “$n_k \epsilon n_0...n_n$” para construir el árbol sintáctico.
- Se asoció “$n_r \epsilon n_0...n_n$”, raíz del árbol sintáctico, al “tree” “t”.
- Se creó un “symbols” “s”.
- Se creó tres instancias de “table”: “vars”, “funcs” y “class”.
- Se asociarón “vars”, “funcs” y “class” a “s”.

Postcondiciones:

- Se creó el conjunto “$r_0...r_n$” de ‘refs” según la nueva sentencia interpretada (creación de objeto).

- Se creó el conjunto "$v_0...v_m$" de 'node" correspondiente a los valores generados en la sentencia interpretada (creación de objeto).
- Se asoció "v_i" a "r_i" como valor de la refercia (creación de enlace).
- Se asoció "$r_i \epsilon r_0...r_n$" a "vars", "funcs" o "class" según la sentencia interpretada (creación de enlace).
- Se actualizó el valor del atributo "canvas" de "s" para reflejar el nuevo estado tras la ejecución de la sentencia.
- Se actualizó el valor del atributo "canvas" de "t" para reflejar el nuevo estado tras la ejecución del la sentencia.

3.2.4. Operación activar_ejecucion_automatica

Nombre: activar_ejecucion_automatica

Responsabilidades: Activa la ejecución automática del proceso de interpretación.

Referencias Cruzadas: Caso de Uso: Activar ejecución automática

Precondiciones:

- Se creó un "tree" "t".
- Se creó un conjunto "$n_0...n_n$" de "node".
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t".
- Se creó un "symbols" "s".
- Se creó tres instancias de "table": "vars", "funcs" y "class".
- Se asociarón "vars", "funcs" y "class" a "s".

Postcondiciones:

- Se establece a "1" el atributo "auto" de "t".

3.2.5. Operación desactivar_ejecucion_automatica

Nombre: desactivar_ejecucion_automatica

Responsabilidades: Desactiva la ejecución automática del proceso de interpretación.

Referencias Cruzadas: Caso de Uso: Desactivar ejecución automática

Precondiciones:

- Se creó un "tree" "t".
- Se creó un conjunto "$n_0...n_n$" de "node".
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t".
- Se creó un "symbols" "s".
- Se creó tres instancias de "table": "vars", "funcs" y "class".
- Se asociarón "vars", "funcs" y "class" a "s".

Postcondiciones:

- Se establece a "0" el atributo "auto" de "t".

3.2.6. Operación limpiar_salida

Nombre: limpiar_salida

Responsabilidades: Limpia la salida producida por el proceso de interpretación

Referencias Cruzadas: Caso de Uso: Limpiar salida

Precondiciones:

- Se creó un "tree" "t".
- Se creó un conjunto "$n_0...n_n$" de "node".
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t".

- Se creó un “symbols” “s”.
- Se creó tres instancias de “table”: “vars”, “funcs” y “class”.
- Se asociarón “vars”, “funcs” y “class” a “s”.

Postcondiciones:

- Se restablece los valores del atributo “canvas” de “t”.

3.2.7. Operación ver_informacion_nodo

Nombre: ver_informacion_nodo

Responsabilidades: Muestra la información de un nodo

Referencias Cruzadas: Caso de Uso: Ver información nodo

Precondiciones:

- Se creó un “tree” “t”.
- Se creó un conjunto “$n_0...n_n$” de “node”.
- Se asoció “$n_i \epsilon n_0...n_n$” a “$n_k \epsilon n_0...n_n$” para construir el árbol sintáctico.
- Se asoció “$n_r \epsilon n_0...n_n$”, raíz del árbol sintáctico, al “tree” “t”.
- Se creó un “symbols” “s”.
- Se creó tres instancias de “table”: “vars”, “funcs” y “class”.
- Se asociarón “vars”, “funcs” y “class” a “s”.

Postcondiciones:

- Se establece los valores del atributo “canvas” de “t” para representar la información del nodo.

3.2.8. Operación ver_tabla_simbolos

Nombre: ver_tabla_simbolos

Responsabilidades: Muestra la información contenida en una tabla de símbolos

Referencias Cruzadas: Caso de Uso: Ver información nodo

Precondiciones:

- Se creó un "tree" "t".
- Se creó un conjunto "$n_0...n_n$" de "node".
- Se asoció "$n_i \epsilon n_0...n_n$" a "$n_k \epsilon n_0...n_n$" para construir el árbol sintáctico.
- Se asoció "$n_r \epsilon n_0...n_n$", raíz del árbol sintáctico, al "tree" "t".
- Se creó un "symbols" "s".
- Se creó tres instancias de "table": "vars", "funcs" y "class".
- Se asociarón "vars", "funcs" y "class" a "s".

Postcondiciones:

- Se establece los valores del atributo "canvas" de "s" para representar la información de la tabla de símbolos.