Informe de implementación del lenguaje L2y3.

En este informe se hablara de los detalles de la implementación del lenguaje L2y3.

Herramientas utilizadas:

* Claire: una herramienta para construir reconocedores de lenguaje
* ESC Java: Una herramienta de verificación de código.

Claire:

La herramienta Claire fue seleccionada para construir el analizador sintáctico y lexicográfico del lenguaje, el motivo de su selección fue que es la única herramienta capaz de generar tanto un analizador lexicográfico como uno sintáctico e integrarlos automáticamente, de esta manera nos fue posible ahorrarnos una gran cantidad de trabajo para lograr esta unión.

La experiencia con la herramienta fue positiva, en el distribuible que se encuentra en la página web del profesor Ascander Suarez, se encontraban múltiples ejemplos los cuales facilitaron el aprendizaje de la herramienta y además de eso existe una documentación adecuada en el wiki de la herramienta la cual permite hacer cualquier consulta fácilmente.

Actualmente toda la construcción del lenguaje de realiza con el uso de esta herramienta y unos cuantos tipos de datos realizados a mano.

ESC Java:

La herramienta ESC Java, nos fue proporcionada para la verificación del código Java, dado que hasta ahora la mayoría del código creado ha sido en Claire, el cual no puede ser analizado por esta herramienta, de momento solo ha sido utilizada para la verificación de los tipos de datos construidos, los cuales debido a su poca complejidad no presentaron error alguno en las verificaciones.

Estado de la implementación.

Diseño del lenguaje:

Se ha terminado el diseño del lenguaje hasta el punto indicado en clase, para mas detalles acerca de esto, por favor consultar el documento del diseño del lenguaje L2y3.

Análisis Lexicográfico:

El análisis lexicográfico se lleva a cabo sin problemas, con excepción de los comentarios que aun no son permitidos en el lenguaje.

Análisis Sintáctico:

El análisis sintáctico se lleva a cabo completamente, con excepción de un conflicto en la gramática que no se ha podido resolver, este conflicto ocurre porque al encontrarse una expresión del tipo: <var1> <operador> <var2 o constante> la gramática no es capaz de identificar si la expresión es booleana o matemática, y por lo tanto tiene problemas en reducirlas apropiadamente.

De momento creemos conveniente realizar una factorización para eliminar este conflicto, sin embargo debido a que este método involucraría realizar bastantes modificaciones a la gramática junto con el hecho de que el error fue descubierto muy tarde, decidimos para esta entrega solamente, eliminar la posibilidad de colocar variables en las expresiones booleanas.

Árbol abstracto:

Actualmente se realiza la construcción del árbol abstracto de tanto expresiones matemáticas como booleanas (con excepción de las expresiones booleanas con variables como se describió anteriormente), también se realiza la construcción del árbol de las instrucciones tipo While, If y For, aunque las dos últimas requieren ser reconstruidas para utilizar nuevos tipos de datos, los cuales a pesar de estar incluidos, no estuvieron listos a tiempo para ser incluidos en esta entrega, el árbol de las subrutinas puede ser creado perfectamente, pero se encuentra limitado por los arboles que no se pueden construir actualmente.

Los demás arboles abstractos para cualquier otra instrucción no mencionada, aun no se han implementado.

Tabla de símbolos:

Actualmente no se está construyendo la tabla de símbolos, pero ya se cuenta con los tipos de datos adecuados para su construcción.

Comprobación de tipos:

Actualmente, el analizador es capaz de diferencias entre una expresión booleana y una matemática a nivel de gramática, dado el hecho de que la tabla de símbolos aun no ha sido implementada, no nos fue posible realizar más comprobaciones de este estilo.

Cualquier otro detalle de la implementación no mencionado anteriormente, seguramente no ha sido comenzado todavía.

Errores conocidos:

En la creación del árbol de una expresión booleana, se utiliza los mismos tipos de datos como si fueran una expresión matemática, esto es una medida provisional y para la próxima entrega se cambiaran estos tipos por los tipos Bool y booleanOperator, incluidos en la entrega.

Actualmente se utiliza el mismo tipo de datos para guardar un Int y un Real, ya existen tipos de datos separados para ambos tipos, pero no dio tiempo de implementarlos para esta entrega.

No existe un tipo de datos para guardar valores de tipo Char, actualmente se utiliza un string, esto será corregido en futuras entregas del lenguaje.