Universidad Tecnológica Nacional
Facultad Regional Santa Fe

SINTAXIS Y SEMÁNTICA

TRABAJO PRÁCTICO 1C2020-Atributos

Tercera parte

Integrantes:

Chort, Julio Alberto – julioch_17@outlook.com

Pacheco Pilan, Federico Ignacio – fedepacheco2112@gmail.com

Reynoso, Valentín – valenreynoso17@gmail.com

Curso:

Comisión ISI "A"

Carrera:

Ing. en Sist. de la Información

Para realizar este trabajo, utilizamos una dinámica diferente (dado a que nos percatamos que usar la de los anteriores trabajos nos era más ineficiente). Nos reunimos vía Discord (como hicimos durante el transcurso de los tres trabajos) pero esta vez uno de nosotros compartía pantalla mientras los otros trabajaban en alternativas y pensaban diferentes atributos. El "principal" compartía su ANTLR a los otros para ir probando que el programa no diera errores sintácticos, mientras los otros dos trabajaban en la búsqueda de atributos que satisfacieran el problema.

Antes de llegar a la táctica mencionada anteriormente, creamos un cuadro con todas nuestras reglas gramaticales y los primeros intentos de atributos:

Regla gramatical	Reglas semánticas
consultas ::= consulta	-
consultas ::= consulta consultas	-

consulta ::= encabezado consumos total_consumos	if (consumos.total_calculado == total_consumos.total_registrado)
	consulta.valido = true;
	else
	consulta.valido = false;
	
	consulta.consumo_maximo = consumos.consumo_maximo;

encabezado ::= línea plan codigo_cliente nombre_cliente rango_fechas	-
línea ::= NUM_LINEA	-
plan ::= PLAN	-
codigo_cliente ::= CODIGO_CLI	-
nombre_cliente ::= NOMBRE_CLI	-
rango_fechas ::= FECHA FECHA	-

consumos ::= registro_consumos	consumos.consumo_maximo = registro_consumos.consumo_maximo;
	
	consumos.total_calculado = registro_consumos.total_calculado;

registro_consumos ::= registro_consumo	registro_consumos.consumo_maximo = registro_consumo.consumo_maximo;
	
	registro_consumos.total_calculado = registro_consumo.total_calculado;

$registro\_consumos_P$ ::= registro_ consumo $registro\_consumos_H$	if (registro_consumo.consumo_maximo > $registro\_consumos_H$.consumo_maximo)
	$registro\_consumos_P$.consumo_maximo = registro_consumo.consumo_maximo;
	else
	$registro\_consumos_P$.consumo_maximo = $registro\_consumos_H$.consumo_maximo;
	
	registro $consumos_P$.total_calculado = registro_consumo.total_calculado + $registro\_consumos_H$.total_calculado;

registro_consumo ::= numero_destino tarifa fecha hora duración importe	registro_consumo.consumo_maximo = importe.valor;
	
	registro_consumo.total_calculado = importe.valor;

numero_destino ::= NUM_LINEA	-

El cuadro anterior no está completo ni están todas las reglas, es sólo la primera parte del mismo.

Este cuadro ayudó bastante en la idealización de los atributos, lo siguiente a eso fue escribir los atributos más sencillos primero (la "inicialización" de los atributos sintetizados) en "importe" y "total_consumos". Desde ese momento, empleamos la táctica explicada al principio de este documento, la cual nos resolvió problemas sintácticos leves (como falta o sobrante de un carácter) y más graves como veremos a continuación.

El programa no tiró ningún error al principio, al ir agregando reglas más complicadas y más arriba empezaron algunos problemas. Por ejemplo, para "registro_consumo registro_consumos_H = registro_consumos" nos faltó hacer el camino del *else* (si el máximo consumo de registro_consumos_H nunca fuera menor que el máximo de registro_consumo, entonces el máximo de registro_consumos nunca tendría un valor asignado y esto llevaría a que el máximo consumo de toda la consulta sea igual a $0.0). Este error nos llevó varios minutos de encontrar, pero afortunadamente agregamos el código faltante y se solucionó el problema de "el consumo máximo es $0.0".

El siguiente gran problema fue la función "*replaceFirst(",",".")*". Al principio, pensamos que era un error de la función "*Float.parseFloat*" pero luego de investigar por internet y hacer pruebas deducimos que funcionaba correctamente. Lo que pasamos por alto nosotros, es que la primera función debía ser asignada a otra cadena (podría ser a ella misma para ahorrar espacio) debido a que lo que hace ese comando es crear otra cadena igual a la anterior, pero reemplazando las comas (,) por los puntos (.). Luego de pensar por un par de horas (entre que no sabíamos muy bien cuál era el error, porque podría ser en cualquier parte del código, y perder tiempo con la función "*Float.parseFloat*") logramos identificar el error y corregirlo.

Luego de solucionar dos errores más importantes y decisivos en el trabajo, lo que quedaba era organizarlo bien y probar con diferentes consultas, las cuales dieron los resultados esperados.

Si tenemos en cuenta las tres entregas (la actual sumada a las dos anteriores), ésta fue para nosotros la más complicada y en la que más hemos invertido nuestro tiempo, más que nada por los errores inexplicables. ANTLR

nos avisaba de errores sintácticos o de funciones que devolvían un “float” que nosotros habíamos interpretado como “int”; pero los problemas a nivel semántico dieron pelea hasta el final.

Dejando todo esto de lado, fue muy interesante ver como los trabajos se iban requiriendo los unos a los otros para formar una gramática totalmente funcional y que acepte e incluso muestre por pantalla algo tan complejo como una consulta telefónica.