Compte rendu réunion technique

Minutes for 17/10/18

Present: Guillaume VANNESSON Nathan BARLOY Quentin MÉRITET Lucas MINÉ

Réunion tenue à Télécom Nancy

Ordre du jour

— Étude de la grammaire donnée dans les spécifications du langage

L'animateur de la séance est Quentin MÉRITET et le secrétaire : Lucas MINÉ. La séance a débuté à 14h00.

Étude de la grammaire donnée dans les spécifications du langage

Le but est de mettre en évidence ce qui gène dans la grammaire pour qu'elle soit LL(1) et d'identifier les récursivités à gauche. On saura par la suite les modifications à effectuer.

Les premiers et les suivant de la grammaire ont été déterminés.

Non terminaux	Premiers	Suivants
program	id nil intList StringLit (- tyId if while for break let	$
dec	type var function	in type var function
tyDec	type	in type var function
ty	tyId array {	in type var function
arrTy	array	in type var function
recTy	{	in type var function
fieldDec	id	,) }
funDec	function	in type var function
varDec	var	in type var function
lValue	id	end ; to do then else , }] infixOp) in type var function $ [; :=
subscript	id	end ; to do then else , }] infixOp) in type var function $ [; :=
fieldExp	id	end ; to do then else , }] infixOp) in type var function $ [; :=
exp	id nil intList StringLit (- tyId if while for break let	end ; to do then else , }] infixOp) in type var function $

seqExp	(	end ; to do then else , }] infixOp) in type var function $
negation	-	end ; to do then else , }] infixOp) in type var function $
callExp	id	end ; to do then else , }] infixOp) in type var function $
infixExp	id nil intList StringLit (- tyId if while for break let	end ; to do then else , }] infixOp) in type var function $
arrCreate	tyId	end ; to do then else , }] infixOp) in type var function $
recCreate	tyId	end ; to do then else , }] infixOp) in type var function $
fieldCreate	id	, }
assignement	id	end ; to do then else , }] infixOp) in type var function $
ifThenElse	if	end ; to do then else , }] infixOp) in type var function $
ifThen	if	end ; to do then else , }] infixOp) in type var function $
whileExp	while	end ; to do then else , }] infixOp) in type var function $
forExp	for	end ; to do then else , }] infixOp) in type var function $
letExp	let	end ; to do then else , }] infixOp) in type var function $

Les règles posant des problèmes de récursivités à gauche sont les suivantes :

— exp / infixExp

— lValue / subscript / fieldExp

Les symboles directeurs sont à faire pour la semaine prochaine.

Fin de la réunion à 15h45.

Next Meeting: Mercredi 24 Octobre à 14h