où r est une référence quelconque qui n'apparaît pas dans e et m et

$$\Sigma(\texttt{\{final T x = t ; p\},e,m,G}) = \Sigma(\texttt{p,e + (x = v),m',G})$$

- Quand l'instruction p est une affectation de la forme x = t ;, si Θ(t,e,m,G) = (v,m') alors

$$\Sigma(\texttt{x = t ;,e,m,G}) = \texttt{(normal,m' + (e(x) = v))}$$

- Quand l'instruction p est une séquence de la forme {p_1 p_2}, si $\Sigma(p_1\texttt{,e,m,G})$ = (normal,m') alors

$$\Sigma(\texttt{\{}p_1\ p_2\texttt{\},e,m,G}) = \Sigma(p_2\texttt{,e,m',G})$$

  et si $\Sigma(p_1\texttt{,e,m,G})$ = (return,v,m') alors

$$\Sigma(\texttt{\{}p_1\ p_2\texttt{\},e,m,G}) = \texttt{(return,v,m')}$$

- Quand l'instruction p est un test de la forme if (b) p_1 else p_2, si Θ(b,e,m,G) = (true,m') alors

$$\Sigma(\texttt{if (b) }p_1\texttt{ else }p_2\texttt{,e,m,G}) = \Sigma(p_1\texttt{,e,m',G})$$

  et si Θ(b,e,m,G) = (false,m') alors

$$\Sigma(\texttt{if (b) }p_1\texttt{ else }p_2\texttt{,e,m,G}) = \Sigma(p_2\texttt{,e,m',G})$$

- Le cas de la boucle reste similaire

$$\Sigma(\texttt{while (b) q,e,m,G}) = \lim_n \Sigma(p_n\texttt{,e,m,G})$$

  où

  p_0 = if (b) giveup ; else skip ;

  et p_{n+1} = if (b) {q p_n} else skip ;.

- Quand l'instruction p est de la forme return t ;, si Θ(t,e,m,G) = (v,m') alors

$$\Sigma(\texttt{return t ;,e,m,G}) = \texttt{(return,v,m')}$$

- Enfin, on ajoute le cas des fonctions, très similaire à celui des fonctions dans la définition de l'évaluation des expressions sauf que si l'objet Σ(p,e'',m'',G) a la forme (normal,m'''), alors on pose

$$\Sigma(\texttt{f(}t_1\texttt{,...,}t_n\texttt{) ;,e,m,G}) = \texttt{(normal,m''')}$$

  et s'il a la forme (return,v,m'''), on pose

$$\Sigma(\texttt{f(}t_1\texttt{,...,}t_n\texttt{) ;,e,m,G}) = \texttt{(normal,m''')}$$

  en ne tenant pas compte de la valeur v : on est dans le cas où une fonction a été utilisée comme une instruction.

Par exemple, quand on exécute l'instruction u = hypothenuse(a,b) ; dans l'environnement e = [a = r_1, b = r_2, u = r_3], la mémoire m = [r_1 = 3.0, r_2 = 4.0, r_3 = 0.0], et l'environnement global G formé de l'environnement e et de la définition de la fonction hypothenuse : (x,y), return Math.sqrt(x * x + y * y) ;, on commence par évaluer l'expression hypothenuse(a,b). Pour cela, on commence par évaluer a, puis b, ce qui produit les valeurs 3.0 et 4.0, sans changer la mémoire. Puis on fabrique un environnement e'' = [a = r_1, b = r_2, u = r_3, x = r_4, y = r_5] et la mémoire m'' = [r_1 = 3.0, r_2 = 4.0, r_3 = 0.0, r_4 = 3.0, r_5 = 4.0]