Première partie
Les actions

Les actions se spécialisent en actions proprement dites (préfixe A) ainsi qu'en actions logiques (préfixe L) qui regroupent des actions et d'autres actions logiques (cf figure 1).

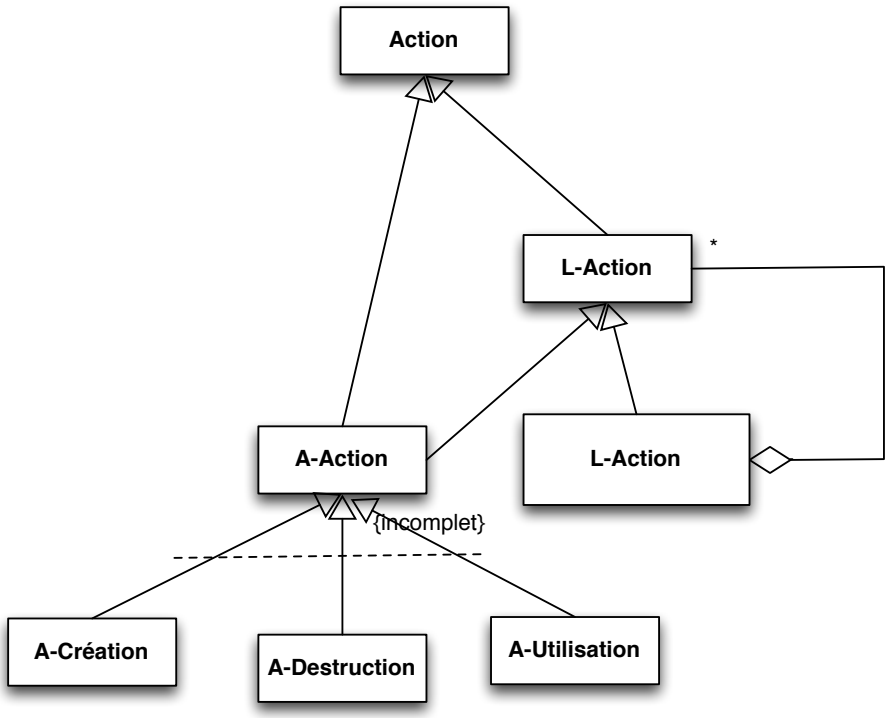

FIGURE 1 – Hiérarchie des actions

1 Les actions A

1.1 Les actions relatives aux objets

1.1.1 A_CreateObject

Cette action crée un objet conformément au processus de fabrication (recette *recipe*) défini au préalable. La signature est A_CreateObject(Created Object)

– **Created Object**, est l'objet à fabriquer

Exemple (pour un agent agriculteur) A_CreateObject(Houe) lance la recette de fabrication de la Houe qui nécessite un bâton pour le manche et un soc (pièce métallique. Si l'agent ne possède pas les deux objets requis pour fabriquer la houe l'action ne fera rien, sinon, elle fabriquera 1 unité de cet objet et l'ajoutera à l'inventaire de l'agent exécutant l'action. Les objets élémentaires ayant servi à la fabrication seront supprimés de l'inventaire.

1.1.2 A_AddObject

Action permettant d'ajouter N objets à l'inventaire de l'agent, sa signature A_AddObject(Change Object, N)

– **Change Object**, précise le type d'objet utilisé pour créer directement l'objet à ajouter à l'inventaire de l'agent (sans utiliser la recette)
– **N**, le nombre d'objets a ajouter

1.1.3 A_DropItem

Action permettant de supprimer N objets de l'inventaire de l'agent, sa signature A_DropItem(Change Object, N).

– **Change Object**, type de l'objet à retirer de l'inventaire de l'agent
– **N**, le nombre d'objets à retirer

1.1.4 A_Transformer

Cette action transforme une ressource prélévée sur un patch en objet, sa signature A_Transformer (Pheromone to Collect, Changed Object). *Remarque : la ressource est appelée phéromone (hérité de Turtle Kit)*

– **Pheromone to Collect**, le type de la ressource que l'agent prélève,
– **Changed Object**, l'objet correspondant (pour le modélisateur) à ajouter à l'inventaire de l'agent

Exemple : un agent exécute A_Transformer (Blé, Meule).

1.1.5 A_UseObject

Cette action correspond à l'usage d'un objet, sa signature est A_UseObject(Changed Object, N)

– **Changed Object**, le type d'objet à utiliser
– **N**, le nombre

1.1.6 A_AddObjectXCogniton

Sa signature est A_AddObjectXCogniton(Changed Object, variation,base, Cogniton).

- **Changed object**, le type d'objet à ajouter,
- **variation**, la variation du nombre d'objets à ajouter,
- **base**, le nombre d'objets à ajouter par défaut,
- **Cogniton**, le cogniton référence.

Cette action, suite à l'existence d'un Cogniton dans l'esprit de l'agent, ajoute des objets à l'inventaire de l'agent en fonction de paramètres contextuel :

- si le cogniton n'existe pas dans l'esprit de l'agent, l'action ajoute *base* objets à l'inventaire,
- si le cogniton existe dans l'esprit de l'agent, l'action ajoute *base* + (*variation* * poids de *Cogniton*) objets à l'inventaire.

Exemple : un agent qui possède le Cogniton Artisan avec un poids de 2, exécute l'action A_AddObjectXCogniton(Houe, 0.5,1, Artisan) ce qui lui permet d'ajouter 1+ 0.5*2 c'est-à-dire 2 Houes à son inventaire ; si l'agent ne possède pas le Cogniton Artisan l'exécution ajoute 1 seule Houe à son inventaire.

1.2 Les actions relatives aux aménagements

1.2.1 A_CreateAmenagement

Cette action

- **Created amenagement**, l'aménagement a construire

L'agent construit l'aménagement **Created amenagement** sur le patch ou il se trouve.

1.2.2 A_EraseAmenagement

- **Created amenagement**, l'aménagement a détruire

L'agent détruit l'aménagement **Created amenagement** sur le patch ou il se trouve.

1.2.3 A_GoToAmenagement

- **Created amenagement**, l'aménagement a cibler

L'agent se dirige vers l'aménagement **Created amenagement** si il en possède un.

1.2.4 A_UseAmenagement

– **Created amenagement**, l'aménagement a utiliser

L'agent utilise **Created amenagement** qui lui appartient si celui ci se trouve sur le même patch que l'agent.

1.3 Les actions relatives aux cognitons

1.3.1 A_AddCogniton

Cette action ajoute un cogniton à l'esprit de l'agent. La signature est A_AddCogniton(Cogniton)

– **Cogniton**,défini le cogniton à ajouter

1.3.2 A_ChangeCognitonWeight

Cette action modifie le poids d'un cogniton. La signature de cette action est A_ChangeCognitonWeight(Cogniton,N)

– **Cogniton**,défini le cogniton à modifier

– **N**,défini le poids du cogniton

1.3.3 A_DoubleCognitons

Cette action ajoute deux cognitons a l'esprit de l'agent. La signature est A_DoubleCognitons(Cogniton1,Cogniton2)

– **Cogniton1**, défini le premier cogniton à ajouter

– **Cogniton2**, défini le deuxieme cogniton à ajouter

1.3.4 A_EmitCogniton

Cette action modifie le poids du cogniton de tout les agents se trouvant sur le même patch que l'éxécutant, sa signature A_EmitCogniton(Cogniton, n)

– **Cogniton**, défini le cogniton à modifier

– **n**, défini le valeur à ajouter

1.4 Les actions relatives aux déplacements

1.4.1 A_AllerVers

Permet à l'agent de se déplacer vers une cible définie plus tôt dans le plan.

1.4.2 A_AvanceAleatoirement

Cette action permet à l'agent d'effectuer un pas dans une direction aléatoire.

1.4.3 A_DoNothing

L'agent ne fais rien.

1.4.4 A_FonderUneVille

Cette action permet à l'agent de se déplacer aléatoirement jusqu'à trouver une zone dépeuplée où créer une nouvelle ville.

1.4.5 A_GetAnotherCityPatch

Cette action définie la cible utilisée dans l'action A_AllerVers comme étant une des villes prise au hasard.

1.4.6 A_GoBackHome

Cette action permet a l'agent de se déplacer d'un pas vers son lieu de création.

1.4.7 A_Move

Cette action déplace l'agent d'un pas dans une direction donnée. Sa signature est A_Move(String)

- **String**, "NORTH","SOUTH","WEST","EAST" défini la direction prise par l'agent

1.4.8 A_SearchForRessources

L'agent cherche autour de lui la ressource donnée et se dirige vers le patch le plus proche et la possédant en plus grande quantité. Si il ne trouve pas de patch contenant cette ressource, il se déplacera aléatoirement. La signature de cette action est A_SearchForRessources(Pheromone To Collect)

- **Pheromone To Collect**, défini la ressource a chercher

1.4.9 A_SmellAndMove

L'agent cherche dans son voisinage immédiat le patch contenant le plus de ressource donnée et fais un pas dans sa direction. Sa signature est A_SmellAndMove(Pheromone To Collect)

- **Pheromone To Collect**, défini la ressource a chercher

1.5 Les actions relatives aux agents

1.5.1 A_Birth

Crée un nouvel agent sur le patch où se situe l'agent éxécutant.

1.5.2 A_Die

Supprime l'agent.

1.5.3 A_DieIfAttributeUnderZero

Supprime l'agent si la valeur de l'attribut descend en dessous de zéro. La signature de cette action est A_DieIfAttributeUnderZero(attributeToCompare)

- **attributeToCompare**, défini l'attribut à vérifier

1.5.4 A_ChangeAttribute

Modifie une caractéristique de l'agent. Sa signature, A_ChangeAttribute(Changed attribute,n)

- **Changed attribute**, défini la caractéristique à modifier
- **n**, défini la valeur à ajouter a la caractéristique

1.5.5 A_CreateGroup

Permet à l'agent de créer un groupe. Sa signature est A_CreateGroup(GroupToCreate)

- **GroupToCreate**, défini le modèle du groupe à créer

1.5.6 A_HireForRole

Permet à l'agent exécutant d'ajouter un agent présent sur le même patch que lui à son groupe . La signature de l'action est A_CreateGroup(GroupToCreate)

- **GroupToCreate**, défini le modèle du groupe dans lequel ajouter l'agent

2 Les actions L

2.1 Les actions logiques de comparaison