

# Construire et exploiter des croyances sur le monde à partir de régularités d'interactions expérimentées

Florian J. Bernard

LIRIS - Université Claude Bernard, Lyon 1  
florian.bernard3@etu.univ-lyon1.fr

**Résumé** Selon Emmanuel Kant, un individu n'a pas accès à la chose en soi tel qu'elle est présente dans le monde, chose que E. Kant appelle noumène, mais construit une représentation interne de cette chose, qu'il appelle phénomène. L'apprentissage développemental donne des directions d'implémentation d'un agent n'ayant pas de connaissances sur son environnement, c'est-à-dire n'ayant pas un accès direct aux noumènes. Il ne possède qu'une liste d'interactions sensorimotrices. Ce mémoire présente une implémentation de la construction de phénomènes à travers le principe de l'apprentissage développemental.

**Abstract.** According to Immanuel Kant, a person does not know the underlying structure of their world "as such" (the noumenal reality), and could only know phenomenal reality (the world "as it appears" through their experience). Developmental learning gives some guidelines to implement an agent which has no knowledge on its environment (i.e. does not have access to the noumenal reality). It only uses sensorimotor interactions. This paper introduces an implementation of the construction of the phenomenon with the precept of developmental learning.

## Table des matières

1	Introduction.....	3
1.1	Modèle incarné.....	4
2	L'Apprentissage Développemental.....	4
2.1	Interactionnisme Radical.....	6
2.2	Interaction.....	7
2.3	Motivation.....	8
2.4	Modèle de programmation d'un agent.....	8
2.5	L'espace.....	9
3	Énoncé du problème.....	10
	Environnement du <i>string problem</i> .....	12
4	Les phénomènes et les tables d'usage.....	13
4.1	Tables d'usages.....	13
	Mise à jour des tables d'usages.....	15
	Table d'usage pour les interactions composites.....	15
4.2	Phénomènes.....	15
	Système décisionnel n°1.....	15
	Système décisionnel n°2.....	19
5	Un environnement pour le développement et l'étude d'agents développementaux.....	21
6	Conclusion.....	24

## 1 Introduction

Ce stage se place dans le contexte de l'apprentissage développemental. Ce domaine met en jeu des agents agnostiques (définis section 1.1). Un agent agnostique ne possède pas d'information sur l'environnement dans lequel il évolue et il n'a pas de comportement pré-enregistré. Il possède des méthodes d'apprentissage pour découvrir ses capacités sensorimoteurs en interagissant avec son environnement. L'agent est incarné et développe des mécanismes cognitifs à travers le couplage de son corps physique et l'environnement [And03,Bro91].

L'apprentissage développemental s'appuie sur le modèle de l'interactionnisme radical qui permet de construire des hiérarchies d'interactions. Ces hiérarchies sont utilisées pour trouver une suite d'interactions permettant de satisfaire l'agent. Le but de cette approche est de construire des comportements augmentant la motivation interactionnelle de l'agent[GMG12]. Néanmoins, l'agent utilisant cette approche ne possède pas de mécanisme lui permettant de représenter les objets du monde.

Comment, à partir d'une suite d'interactions ne possédant pas de sémantique, un agent agnostique peut-il construire une représentation des « choses » de son environnement ?

La phénoménologie se présente comme l'une des solutions envisageables pour permettre à un agent d'acquérir des connaissances à partir de l'expérience réalisée dans un environnement. La phénoménologie est un courant philosophique qui se concentre sur l'étude des phénomènes, de l'expérience vécue et des contenus de conscience.

La phénoménologie a été introduite par Edmund Husserl en 1913. Elle permet, d'un point de vue philosophique, de définir la conscience des choses ainsi : toute conscience doit être conçue comme conscience de quelque chose.

*« On ne trouve dans la donnée immédiate [de la conscience] rien de ce qui, dans la psychologie traditionnelle, entre en jeu, comme si cela allait de soi, à savoir : des data-de-couleur, des data-de-son et autres data-de-sensation ; des data-de-sentiment, des data-de-volonté, etc. Mais on trouve ce que trouvait déjà René Descartes, le cogito, l'intentionnalité, dans les formes familières qui ont reçu, comme tout le réel du monde ambiant, l'empreinte de la langue : le « je vois un arbre, qui est vert ; j'entends le bruissement de ses feuilles, je sens le parfum de ses fleurs, etc. » ; ou bien « je me souviens de l'époque où j'allais à l'école », « je suis inquiet de la maladie de mon ami », etc. Nous ne trouvons là, en fait de conscience, qu'une conscience de... » [Hus76]*

Selon Emmanuel Kant, pour avoir une conscience de quelque chose, l'individu utilise une représentation interne des « choses » du monde qu'on appellera phénomène par la suite. Ces « choses » peuvent représenter un objet physique, une relation entre plusieurs objets ou une entité abstraite. A contrario, le noumène désigne la « chose » tel qu'elle est dans le monde. Le noumène en lui-même est inaccessible à l'individu qui devra utiliser des phénomènes pour se le représenter.

Dans ce stage, nous avons réalisé un agent implémentant les 4 principes ci-dessus, à savoir :

— le concept de l'apprentissage développemental ;

- le modèle de l’interactionnisme radical ;
- la phénoménologie d’Edmund Husserl ;
- la distinction entre les noumènes et les phénomènes.

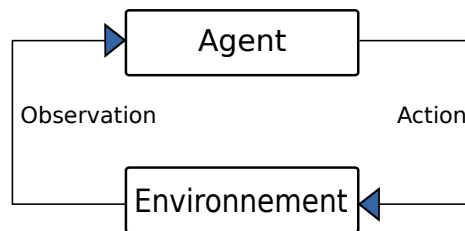
L’agent découvre des régularités de son environnement, à travers ses interactions, et les utilise pour construire des phénomènes. Pour visualiser l’état de la mémoire de l’agent, j’ai développé un logiciel gérant la simulation d’agent avec différents algorithmes, un système de visualisation d’interactions et des phénomènes avec la possibilité de tester les agents dans différents environnements.

La prochaine partie comporte une présentation/état de l’art de l’apprentissage développemental et du modèle de l’interactionnisme radical. La seconde partie explicite la création des phénomènes et des états de croyance. La fin du rapport présente le logiciel de simulation pour finir sur la conclusion.

### 1.1 Modèle incarné

Les propriétés clés du modèle incarné peuvent être résumées par la notion d’inversion du cycle d’interaction illustré en n°2 [GC14,VT], par contraste avec le cycle traditionnel présenté en figure n°1.

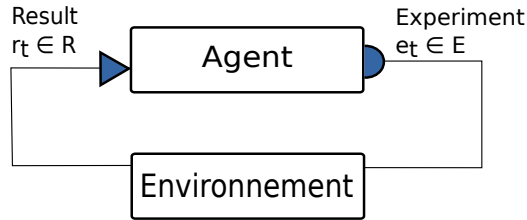
Un agent conçu avec le modèle incarné est dit « agnostique » car il n’a pas de connaissance prédéfinie du monde, et ses données d’entrée (rt) ne représentent pas l’état du monde. Non seulement un agent agnostique n’est pas markovien, mais il n’est pas non plus « partiellement markovien » (à la différence des modèles « Partially Observable Markov Decision Process » (POMDP) [CKL94]).



**FIGURE 1.** Cycle observation/action traditionnel : les points de début et de fin du cycle ne sont traditionnellement pas explicités. Conceptuellement, le cycle commence par le fait que l’agent reçoit une observation qui est une représentation partielle du monde (possiblement bruitée, symbolique ou non), ensuite, l’agent décide d’une action sur la base de cette observation.

## 2 L’Apprentissage Développemental

L’apprentissage développemental a été, dans un certain sens, introduit par Jean Piaget dans les années 50, lorsqu’il travaillait sur la psychologie développementale [Pia59]. En dépit du fait, que les travaux de Piaget ont été menés dans



**FIGURE 2.** Cycle expérience/Résultat. Le cycle d’interaction commence par le fait que l’agent initie une expérience dans l’environnement (début conceptuel du cycle : demi-cercle), et se termine par le fait qu’il reçoit un résultat  $r_t$  de cette expérience (fin conceptuelle du cycle : triangle). Le résultat  $r_t$  n’est pas une représentation du monde, car, dans un état du monde donné, l’agent peut recevoir différents résultats en fonction de l’expérience qu’il fait. L’agent doit donc construire sa propre représentation du monde (perception construite) au lieu de la recevoir directement comme c’est le cas dans les modèles traditionnels tels que celui de la figure n°1

les années 50, l’apprentissage développemental est un nouveau courant de l’intelligence artificiel. Il sollicite des connaissances de plusieurs disciplines comme la robotique, les neurosciences, la psychologie et les sciences cognitives. En apprentissage développemental, les agents :

- ne possèdent pas d’ontologie présupposée (i.e : pas de connaissances présupposées sur le monde) ;
- sont non-markovien, ni même partiellement markovien ;
- construisent leur propre représentation de l’environnement qui par définition sera différente de celle que nous pouvons avoir ;
- utilisant le concept d’auto-programmation<sup>1</sup> [TNSW13] ;
- dispose de possibilités d’interactions sensorimotrices primitives prédéfinies.

Le but de l’agent est d’apprendre, découvrir, organiser et exploiter des régularités d’interactions dans le temps et l’espace pour favoriser certains comportements et améliorer son adaptabilité.

Un point important de l’apprentissage développemental est que l’agent n’est pas un observateur passif de son environnement, il ne réagit pas à un stimuli donné par son environnement, mais choisit volontairement ce qu’il va observer ou réaliser au travers de ses interactions. Les interactions sont les éléments centraux de l’agent, elles lui permettent d’interagir avec son environnement et de construire ses connaissances. La modélisation des interactions est un point important pour la conception de l’agent.

Nous avons choisi d’utiliser le modèle de l’interactionnisme radical qui permet de représenter les actions et les résultats sous forme d’interaction.

1. L’auto-programmation est définie comme étant la possibilité de créer un code exécutable qui peut être exécuté au besoin cf. : <http://liris.cnrs.fr/ideal/mooc/lesson.php?n=041>

## 2.1 Interactionnisme Radical

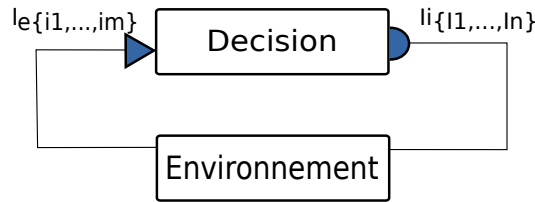
Le modèle de l'interactionnisme radical [GA13] est une façon de traduire les idées théoriques de l'apprentissage développemental [Pia59,WF13,O'R11] dans les algorithmes. Dans les modèles cognitifs, l'agent interagit et construit ses connaissances avec son environnement par le biais de ses propres expériences et de leurs résultats. Ce modèle utilise comme primitive les interactions et non le couple expérience/résultat. Les interactions sont développées dans la partie 2.2

Le modèle est décrit avec la figure n°3. À l'initialisation, l'agent possède une liste prédéfinie d'interactions sensorimotrices  $I$ , appelées **interactions primitives**. À chaque cycle de décisions, l'agent choisit une interaction appelée **interaction intended** qu'il tentera de réaliser dans l'environnement. L'interaction effectivement réalisée dans l'environnement, appelée **interaction enacted** est retournée à l'agent. Les interactions *intended* et *enacted* appartiennent à la même liste d'interactions sensorimotrices  $I$ .

Il est à noter que l'agent ne connaît pas les conséquences des différentes interactions ni sur l'environnement ni sur lui-même et des liens qui peuvent exister entre chaque interaction. De plus, nous ne souhaitons donner aucune heuristique qui pourrait l'aider à comprendre ces interactions.

L'environnement représente le couplage entre le corps de l'agent et le monde. Les mécanismes mis en place dans le système décisionnel permettent :

- d'apprendre des interactions sur plusieurs niveaux tout en évitant l'explosion combinatoire ;
- de sélectionner la prochaine interaction en fonction du contexte d'interaction précédemment réalisée.



**FIGURE 3.** Modèle de l'interactionnisme radical

L'agent étant agnostique [GS12], c'est-à-dire, qu'il ne connaît pas du tout l'environnement, il interagit avec celui-ci uniquement aux travers des interactions qu'il possède. L'agent découvre des suites d'interactions qui sont avantageuses de son point de vue.

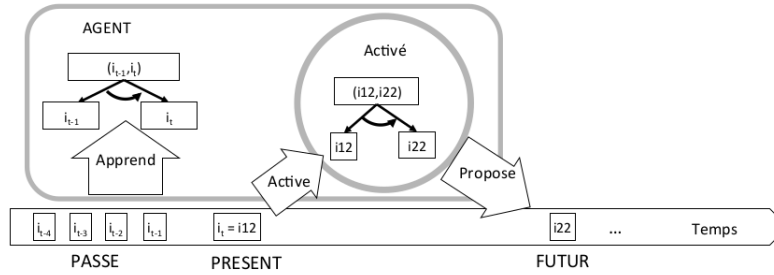
Le système de décision ne commence pas par une observation donnée par l'environnement mais par une interaction sélectionnée par l'agent. L'agent est donc pro-actif, il sélectionne la prochaine interaction qu'il souhaite *enact* en prenant en compte l'expérience passée de l'agent et non une fonction de l'environnement.

## 2.2 Interaction

Dans le contexte des interactions, on ne peut pas dissocier une perception d'une action. Elle est composée d'un couple abstrait action/résultat défini *a priori* qui est indépendant de l'environnement, du système sensoriel et des actionneurs que possède l'agent. Elle n'a pas de sémantique et ne définit pas un état du monde. Les interactions caractérisent le *couplage* entre l'agent et l'environnement. Les algorithmes mis en place dans le système décisionnel doivent prendre en compte le fait que la connaissance de l'agent se construit uniquement à travers les interactions.

Pour que l'agent puisse apprendre et exploiter ses connaissances, l'algorithme de décision permet d'apprendre des interactions composites sur plusieurs niveaux. Celles-ci donnent à l'agent la possibilité de développer des comportements de plus haut niveau.

Une **interaction composite** est composée d'une pré-interaction et d'une post-interaction (qui elles mêmes peuvent-être composites ou primitives). La pré-interaction renseigne sur le contexte dans lequel l'agent se trouve. La figure n°4 indique le processus d'apprentissage et de sélection des interactions composites.



**FIGURE 4.** Principe d'apprentissage et de sélection d'une interaction composite. Afin de déterminer la prochaine interaction, la décision sélectionne toutes les interactions composites ayant comme pré-interaction l'interaction précédemment *enacted* {i12}. Puis, analyse l'interaction composite qui a le plus de chance d'être *enact* avec la plus forte motivation interactionnelle. L'interaction composite ainsi sélectionnée a comme pré-interaction l'interaction *enacted* {i12} et comme post-interaction l'interaction à réaliser.

Pour pouvoir réaliser cet apprentissage, l'agent doit pouvoir :

- décomposer une interaction composite *intend* ;
- reconstruire une interaction composite *enact*.

Une interaction composite est exécutée en une seule fois. Elle peut réussir ou échouer sur chaque interaction primitive qu'elle possède.

La figure n°5 est une sous partie d'un flux d'interactions réalisées par un agent utilisant les interactions composites. La première interaction a été correctement

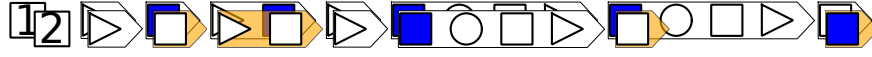


FIGURE 5. Interactions *intended* (ligne 1) et *enacted* (ligne 2) d'un flux d'interaction.

prévu par l'agent. À l'inverse la deuxième a échoué car l'interaction *intended*  $\{\blacksquare\}$  est différente de l'interaction *enacted*  $\{\square\}$ . L'interaction composite *intended*  $\{\triangleright, \blacksquare\}$  a échoué sur la post-interaction  $\{\square\}$ .

Une **interaction alternative** est une interaction qui peut-être *enact* à la place d'une interaction *intended*. Dans la figure n°5, l'interaction  $\{\square\}$  est une interaction alternative à l'interaction  $\{\blacksquare, \circ, \square, \triangleright\}$ . Cette propriété n'est pas symétrique.

Lorsque l'alternative est symétrique, c'est-à-dire que deux interactions sont alternatives l'une de l'autre alors on parle d'**interaction opposée** et de **régularité immédiate**. Les interactions  $\{\square\}$  et  $\{\blacksquare\}$  sont des interactions opposées l'une à l'autre.

Afin que l'agent puisse sélectionner une interaction plutôt qu'une autre, une motivation permet de les différencier.

### 2.3 Motivation

Pour que l'agent puisse déterminer les interactions intéressantes, le concept de motivation interactionnelle [GMG12] est mis en place dans l'apprentissage développemental. Ce concept permet de définir des interactions *plaisantes* et *déplaisantes* pour l'agent. Grâce à cette motivation l'agent sélectionne une interaction sans autre but que la satisfaction qu'elle peut lui procurer [Ste04]. Elle est implémentée grâce à une valence définie *a priori*. Si la valence est positive (resp. négative), l'interaction est plaisante (resp. déplaisante). Si la valence est nulle, l'interaction en elle même n'apporte rien à l'agent, par contre couplée avec d'autres interactions, celle-ci peut amener aux interactions plaisantes.

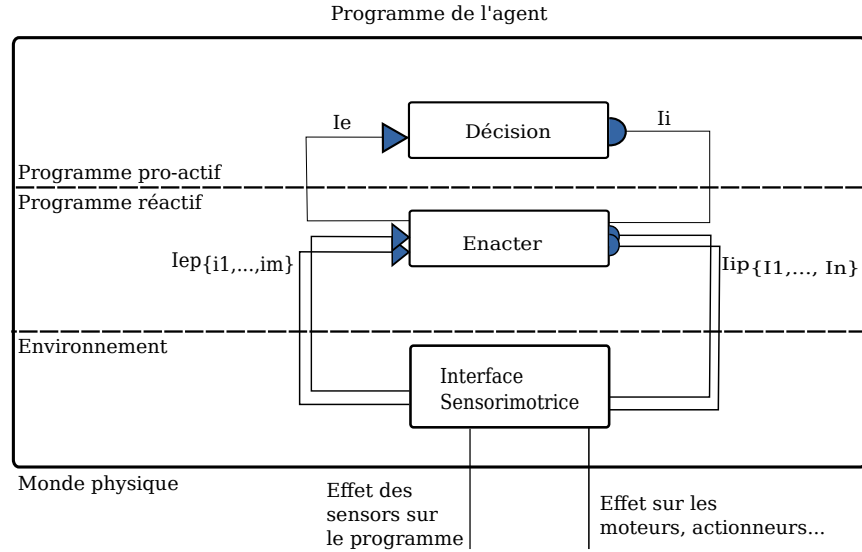
D'autres formes de motivations sont également prises en compte et permettent de construire des comportements différents :

- l'agent aime prévoir les prochaines interactions à *enact* ;
- l'agent peut être « curieux » : l'agent va préférer tester les liens entre les différentes interactions pour avoir le plus de choix possibles, et potentiellement avoir la meilleure approche ;
- l'agent peut être « joueur » : l'agent va répéter des interactions dans le but de les réutiliser par la suite dans un autre contexte.

### 2.4 Modèle de programmation d'un agent

La figure n°6 explicite l'implémentation d'un agent avec le modèle de l'interactionnisme radical. La partie pro-active possède les mécanismes d'apprentissage et de sélection des interactions. Elle sélectionne une interaction (primitive ou





**FIGURE 6.** Schéma de l'implémentation d'un agent

composite)  $I_i$  qui sera réalisé à l'aide de la partie réactive. L'*enacteur* décompose, si besoin, l'interaction *intended*  $I_i$  en interaction primitive  $I_{ip_1} \dots I_{ip_n}$  et l'exécute à travers les différentes interfaces sensorimotrices. Les résultats des capteurs se présentent sous la forme d'interactions primitives *enacted*  $I_{ep_1} \dots I_{ep_m}$ . La taille de l'interaction composite *enacted* peut être inférieure ou égale à la taille de l'interaction *intended*. L'*enacteur* reconstruit l'interaction composite *enacted*  $I_e$  et la retourne à la décision qui va pouvoir apprendre de nouvelles interactions. Dans la suite du mémoire, la partie pro-active de l'agent sera mise en avant. Nous considérons que le programme réactif et l'environnement sont fonctionnels.

## 2.5 L'espace

Mes travaux se basent sur la thèse de Simon Gay [Gay14] et sur le MOOC réalisé par Olivier Georgeon<sup>2</sup>. Tous deux ont travaillé sur le modèle de l'interactionnisme radical. S. Gay s'est orienté sur la représentation de l'espace et des objets que l'environnement permet d'afforder [Gib77]. « Une affordance est définie comme une possibilité d'interaction proposée par l'environnement à un agent. En effet, dans ces expériences, chaque objet révèle qu'il peut être saisi par un certain mouvement, et, de ce fait, afforde ce mouvement. ». Durant sa thèse, S. Gay a grandement travaillé sur la représentation de l'espace dans un agent implémentant l'apprentissage développemental et la perception d'objet dans l'espace observable et de l'espace global.

La représentation des objets est réalisée avec la *signature d'interaction*. Une signature d'interaction est représentée par un ensemble d'interactions dont le

2. <http://liris.cnrs.fr/ideal/mooc/>







succès caractérise la présence ou l'absence d'un objet affordé par les interactions. Si des interactions possèdent les mêmes signatures alors celles-ci relatent du même objet.

Les mécanismes de décision se servent de ces signatures pour effectuer les interactions satisfaisantes.

Son travail a été une source d'inspiration pour la réalisation de ce stage. La représentation des *objets* du monde met en place une partie des mécanismes associés aux signatures.

### 3 Énoncé du problème

Le problème est défini en se mettant à la place de l'agent (n.b. : L'environnement est présenté de manière plus conventionnelle au paragraphe 3). Nous ne connaissons rien de l'environnement et nous devons créer une structure permettant de représenter les connaissances construites par l'agent. Les interactions de la figure n°7 représentent les perceptions et les actions de l'agent. À partir du flux d'interactions de la figure n°8, nous souhaitons que l'agent découvre des régularités qu'il puisse exploiter.

 0	 1	 0
 0	 -1	 0

**FIGURE 7.** Liste des interactions que possède l'agent dans l'environnement inconnu.

À partir du flux d'interactions de la figure n°8 nous pouvons trouver :

- les régularités immédiates ;
- des régularités séquentielles ;

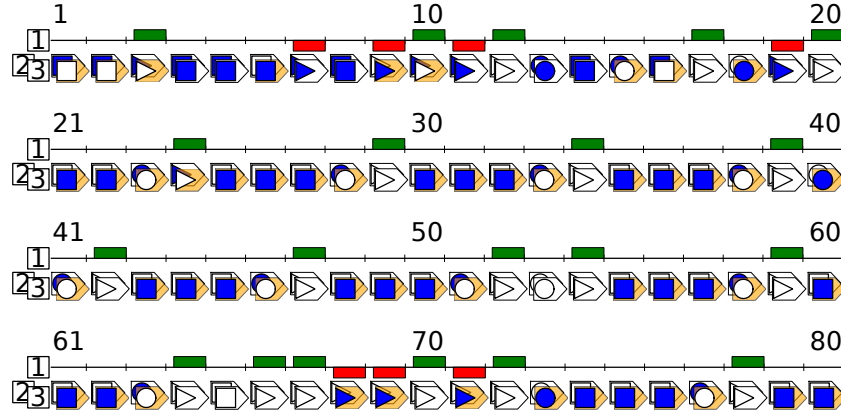
Les régularités immédiates sont données grâce aux lignes n°2 et 3. Au début du flux d'interaction, l'agent *intend* des interactions sans savoir si elles sont réalisables. Régulièrement, l'interaction *enacted* ne correspond pas, ce qui implique que cette interaction est une interaction alternative. Au pas 1, l'interaction {■} a pour alternative {□}. Au pas 6, les interactions {■} et {□} deviennent des interactions opposées l'une à l'autre.

Les régularités séquentielles de niveau 2 que l'on peut trouver via ce flux sont représentées par la figure n°9. Ces régularités sont définies par l'environnement, et nous posons l'hypothèse qu'avec ces régularités, l'agent peut construire des connaissances sur l'environnement.

La régularité d'interactions {□}, {□} informe sur la présence d'une observation. Cette interaction est supposée répétable indéfiniment. Elle est considérée

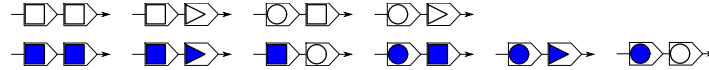
---

4. <http://liris.cnrs.fr/ideal/mooc/lesson.php?n=058>



**FIGURE 8.** Flux d'interactions réalisé par l'algorithme d'O. Georgeon disponible sur le MOOC<sup>4</sup>. La ligne n°1 représente la valence qui caractérise la motivation interactionnelle. En vert, la valence est positive, l'interaction est plaisante. En rouge l'interaction est déplaisante, la valence est négative. L'absence de symbole indique que l'interaction n'apporte rien. La ligne n°2 est l'interaction *intended* et la ligne n°3 est l'*enacted*. Les interactions en orange ont été mal prévues par l'agent qui est alors insatisfait. L'agent a bien prédit les interactions *enacted* blanche par conséquent son humeur est satisfaite.

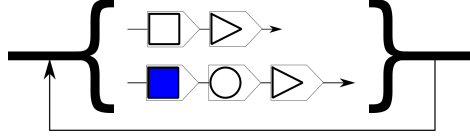
comme étant une **interaction persistante** qui renseigne sur la présence ou l'absence d'un objet. Les interactions  $\{\bullet\}$ ,  $\{\triangleright\}$  et  $\{\blacktriangleright\}$  ne se répètent pas de manière systématique, on considère que ce sont des interactions qui modifient l'environnement et sont appelées **interactions sporadiques**. L'interaction  $\{\circ\}$  ne donne pas d'informations particulières sur son état, par contre elle renseigne des interactions qui peuvent être effectuées juste après elle.



**FIGURE 9.** Régularités observées dans le flux d'interaction. Ces régularités sont tirées du flux d'interaction de la figure n°8. Elles sont liées aux interactions *enacted*. Par exemple, la régularité n°3 est réalisée au pas 15.

L'agent est motivé par la réalisation de l'interaction  $\{\triangleright\}$ , de ce fait, nous espérons pouvoir construire les régularités séquentielles de la figure n°10. Ces régularités sont déjà apprises par l'algorithme via les interactions composites mais ces interactions ne permettent pas d'avoir une représentation des *objets* de l'environnement.

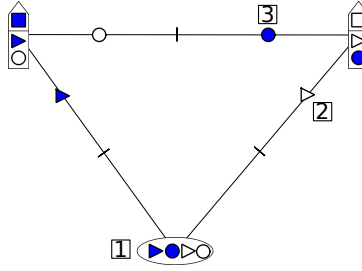
Les interactions sporadiques  $\{\circ\}$  et  $\{\bullet\}$  permettent une fois *enacted* de réaliser respectivement les interactions persistantes  $\{\square\}$  et  $\{\blacksquare\}$ . Nous appelons **état de croyance** l'information qui permet à une interaction sporadique de prédire



**FIGURE 10.** Régularités séquentielles que l'on souhaiterait que l'agent trouve et utilise pour satisfaire sa motivation

une interaction persistante. L'interaction  $\{\square\}$  permet d'*enact* l'interaction  $\{\triangleright\}$ . L'interaction  $\{\circ\}$  permet de changer l'environnement de sorte que l'interaction persistante  $\{\square\}$  soit toujours *enact*. De ce fait, l'interaction  $\{\circ\}$  possède comme état de croyance l'interaction persistante  $\{\square\}$ .

Avec ces informations et en considérant que l'interaction  $\{\circ\}$  est par défaut une interaction sporadique, nous pouvons créer le graphe de Pétri de la figure n°11 qui représente les liens entre les interactions persistantes et sporadiques.



**FIGURE 11.** Graphe de Pétri que nous souhaitons que l'agent construise par rapport aux régularités de niveau 2 de la figure n°9. Le nœud n°1 représente l'état de croyance *unknown* qui regroupe les interactions sporadiques. Les autres nœuds représentent les interactions considérées comme persistantes. La partie inférieure des nœud représente les interactions qui pourront être réalisées lorsque l'agent se trouve dans cet état de croyance. Lorsque l'agent est dans l'état de croyance  $\langle \square \rangle$  et qu'il réalise l'interaction  $\{\triangleright\}$  alors l'état de croyance deviendra *unknown*. Ce lien est représenté par le n°2. Le n°3 représente un état de croyance associé à l'interaction sporadique  $\{\bullet\}$ .

**Environnement du *string problem*** L'environnement décrit avec les *yeux* de l'agent est le *string problem* [GH13]. Cet environnement est une chaîne de nombres : 1-7-3-2-9-3-5-6-7-8-1-2-4-0-9-8-5-4-6-0, que l'agent doit trier. L'agent est positionné sur un nombre et peut :

- avancer vers un nombre plus grand (resp. plus petit) avec l'interaction  $\{\square\}$  (resp.  $\{\blacksquare\}$ );

- échanger le nombre sur lequel il se situe avec celui devant lui. Si après l'échange le nombre devant est plus grand (resp. plus petit) alors l'interaction  $\{\circ\}$  (resp.  $\{\bullet\}$ ) est *enact* ;
- avancer sur un nombre plus grand (resp. plus petit) via l'interaction  $\{\triangleright\}$  (resp.  $\{\blacktriangleright\}$ ).

De notre point de vue, l'agent est motivé à trier la chaîne de caractères pour, par la suite, réaliser l'interaction  $\{\triangleright\}$ . Pour lui, la chaîne de caractères n'existe pas et pour le moment évolue dans un environnement qui donne du plaisir lors de l'*énaction* de l'interaction  $\{\triangleright\}$ . Cet environnement est volontairement simple pour rendre accessible l'analyse de la mémoire de l'agent.

De ces constats, je propose une structure permettant à l'agent d'obtenir une représentation des objets à travers ses interactions primitives.

## 4 Les phénomènes et les tables d'usage

### 4.1 Tables d'usages

Les tables d'usage sont inspirées du travail que S. Gay a réalisé lors de sa thèse et qu'il appelle *signature*. Chaque interaction primitive possède une table d'usage. Elles sont définies par la figure n°12.

Une récompense est associée par chaque table d'usage qui correspond pour une interaction :

- persistante : à la valence de la meilleur interaction confirmé ;
- sporadique : à la valence de l'interaction ;
- sporadique avec croyance : à la valence de l'interaction plus la moitié de l'état de croyance.

On peut déduire de la table d'usage de la figure n°12 que l'interaction  $\{\circ\}$  n'est pas répétable car la table des pré-interactions indique qu'elle a été *intended* 2 fois, mais jamais *enacted*. L'interaction  $\{\square\}$  n'est pas réalisable avant l'interaction  $\{\circ\}$  car celle-ci a été *intended* 2 fois sans jamais être *enacted*.

Toujours avec la table des pré-interactions, les interactions  $\{\blacksquare\}$  et  $\{\bullet\}$  ont toujours été *enacted* avec succès, ce sont donc des interactions qui permettent d'*enact* l'interaction  $\{\circ\}$ . Elles sont les conditions qui permettent d'*enact* de manière *certaine* (i.e. du point de vue de l'agent) l'interaction  $\{\circ\}$ .

De la même manière avec la table des post-interactions, l'interaction  $\{\circ\}$  permet d'*enact*  $\{\square\}$ ,  $\{\bullet\}$  et  $\{\triangleright\}$ . Elles sont appelées d'interactions confirmées par l'interaction  $\{\circ\}$ .

L'interaction  $\{\bullet\}$  est une interaction opposée et qui a été *enacted* trois fois à la place de l'interaction  $\{\circ\}$ .

Avec cette interaction nous pouvons construire le sous graphe de Pétri de la figure n°13 qui reprend les points ci-dessus.



**Mise à jour des tables d’usages** Lorsque l’*enacter* (cf. n°6) à réaliser l’interaction *intended* et a recréé l’interaction *enacted*, il met à jour les tables d’usages des deux interactions à travers la structure appelée *Phénomènes*.

La mise à jour est effectuée en utilisant le principe suivant :

- l’interaction *enacted* doit mettre à jour les poids de succès ou de l’échec des pré-interactions *intended* et *enacted* ;
- l’interaction précédemment *enacted* met à jour les poids de succès ou d’échec des post-interactions *intended* et *enacted* ;
- Si l’interaction *intended* est différente de l’interaction *enacted* alors l’interaction *intended* doit mettre à jour le poids d’échec de la précédente interaction *enacted* ;
- Si l’interaction *intended* est égale à la précédente interaction *intended* mais que les interactions *enacted* sont différentes alors l’interaction *intended* est considérée comme sporadique.

**Table d’usage pour les interactions composites** Pour le moment, les interactions composites sont gérées de la même manière que les interactions primitives mais n’apportent pas d’information pertinente dans la représentation des objets de l’environnement. Par conséquence, elles ne seront pas détaillées dans ce mémoire.

## 4.2 Phénomènes

*Nota bene* : dans le domaine de la phénoménologie, les objets du monde se représentent non pas par l’objet en lui-même mais par les interactions que l’individu peut effectuer sur l’objet. Les composantes du monde sont inaccessibles à l’individu, mais celui-ci arrive à distinguer les objets *rond* et *carré* facilement. Le fait de voir un objet active dans le cerveau une sorte de simulation des interactions que nous pouvons effectuer avec l’objet. Cette représentation nous donne la possibilité de comprendre que deux objets sont identiques dans des situations différentes [MFF<sup>+</sup>97]. Pour permettre la gestion des phénomènes, nous avons implémenté 2 algorithmes.

**Système décisionnel n°1** Ce système décisionnel possède deux phases. L’une pour l’apprentissage des phénomènes et la seconde pour l’exploitation et la mise à jour des connaissances. Les interactions sont initialisées comme étant inconnues. L’agent va répéter, jusqu’à un certain seuil, une interaction considérée comme inconnue, pour déterminer si elle est persistante ou sporadique. Le flux d’interactions généré par cet algorithme dans le *string problem* est représenté par la figure n°14.

L’algorithme n°1 sélectionne une interaction *intended*. Les différentes humeurs permettent à l’algorithme de différencier l’apprentissage de l’exploitation. L’humeur *excited* est utilisée pour reproduire la précédente interaction *intended* afin de déterminer si cette interaction est persistante ou sporadique. Elle sera

```

1 Believe ← "unknown"
2 mood ← "curious"
3 Function getIntend()
4   si mood = "curious" alors
5     | intended ← leastTriedExperiment(Believe)
6   sinon
7     si mood = "hedonist" alors
8       | intended ← intendedMaxExpectedOutcomeValence(Believe)
9     sinon
10      si mood = "excited" alors
11        | intended ← lastEnacted
12      fin
13    fin
14  fin
15  retourner intended

```

**Algorithme 1** : Algorithme de sélection de l'interaction *intended*

déterminée dans l'algorithme n°2. L'humeur *curious* favorise, dans le contexte de l'interaction *enacted*, la réalisation de test sur les post-interactions les moins utilisées. L'*hedonist* est le comportement par défaut. Il sélectionne l'interaction qui à le plus de chance d'être *enacted* et qui privilégie la motivation interactionnelle. La sélection de cette interaction est détaillé dans l'algorithme n°3

L'algorithme n°2 permet de sélectionner l'humeur de l'agent en fonction de l'interaction *enacted*. Si l'agent était excité, alors il compare la précédente interaction *enacted* avec la nouvelle :

- si les interactions sont identiques et que le seuil d'excitation n'est pas atteint alors l'agent est toujours excité par cette interaction ;
- si les interactions sont identiques et que le seuil d'excitation est atteint alors l'interaction est considérée comme persistante ;
- si les interactions sont différentes alors l'interaction *enacted* est sporadique.

Ensuite l'agent met à jour son état de croyance développé par la fonction *updateAndGetBelieve*. Après, si l'interaction *enacted* est inconnue, alors l'agent va être excité et va donc répéter cette interaction pour déterminer si elle persistante ou sporadique. Si la table des post-interactions comporte des interactions non testées dans le contexte de l'interaction *enacted*, alors l'agent va être curieux et expérimenter une interaction jamais *intend* et *enact*. Sinon il devient hédoniste.

La fonction *updateAndGetBelieve* permet de mettre à jour l'état de croyance de l'interaction *enact*. Si elle est persistante, alors l'état de croyance sera l'interaction elle-même. Si l'interaction sporadique possède une post-interaction persistante et qui est confirmé, alors l'état de croyance sera la post-interaction persistante, sinon il sera inconnu.

L'algorithme n°3 sélectionne l'interaction la plus intéressante dans le contexte de l'état de croyance courant :

- si l'état de croyance existe : sélectionne une interaction satisfaisante ou une interaction amenant a une interaction satisfaisante ;



- l'interaction est sporadique : retourne une interaction persistante qui a le plus de chance d'être *enact*.

```

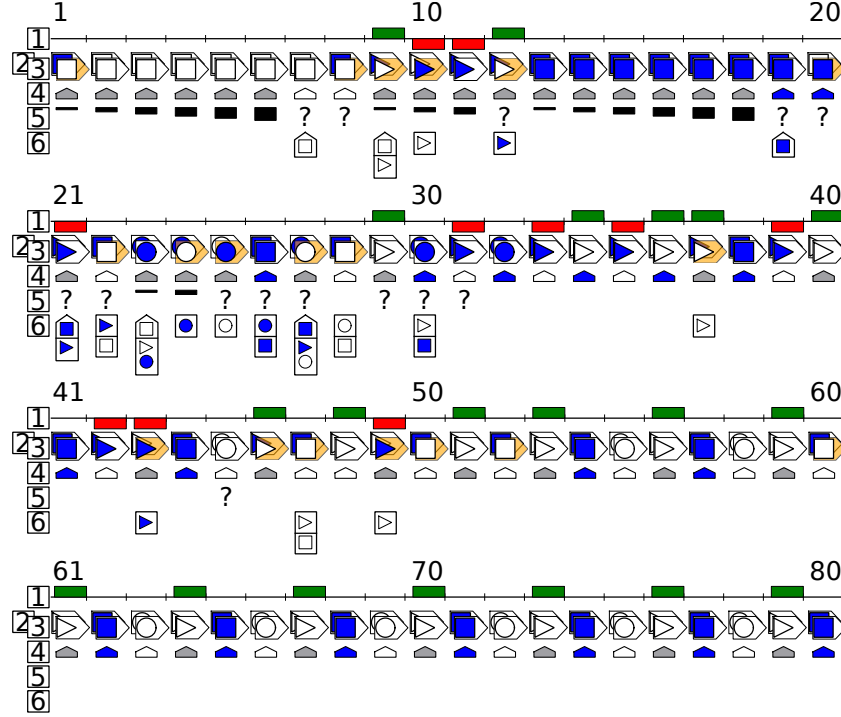
1 Procedure hasEnacted(Interaction enacted)
2   si mood = "excited" alors
3     si enacted ≠ lastEnacted alors
4       | enacted.setUnpersistent()
5     sinon
6       | si excitement > excitementThreshold alors
7         | enacted.setPersistent()
8       | sinon
9         | excitement ++
10      fin
11   fin
12 fin
13 Believe ← updateAndGetBelieve(intended, enacted)
14 si mood ≠ excited alors
15   | mood ← "hedonist"
16 fin
17 si enacted.isUnknown() alors
18   | mood ← "excited"
19   | lastEnacted ← enacted
20 sinon
21   | si all the experiments have not been tried yet in the context of believe
22     | alors
23       | mood ← "curious"
24   fin
25 fin
26 retourner

```

**Algorithme 2 :** Mise à jour de la décision lors de la réalisation d'une interaction

---

6. Une expérience correspond au test d'enactage d'une interaction ou de l'une de ses opposées.



**FIGURE 14.** Flux d'interaction de l'algorithme d'apprentissage et d'exploitation. La ligne n°4 représente l'état de croyance dans lequel se trouve l'interaction *enacted*. En gris, l'état de croyance est inconnue, en blanche elle fait référence à l'état de croyance  $\langle \square \rangle$  et en bleu à l'état de croyance  $\langle \blacksquare \rangle$ . La ligne n°6 montre la création et la mise à jour des interactions persistantes et sporadique. L'humeur de l'agent est sur la ligne n°5. Du pas 1 à 6, l'excitation de l'agent augmente jusqu'au seuil (prédéfini). Au pas 7, l'interaction  $\{\square\}$  est considérée comme persistante. Au pas 8, l'agent est curieux car il connaît l'interaction persistante  $\{\square\}$ , mais sa tables d'usage est vide. Il va alors essayer une expérience<sup>6</sup> qui permettra de remplir une partie de la tables d'usage. Au pas 9, l'agent est excité par l'interaction *enacted*  $\{\triangleright\}$ , mais au pas 10, cette interaction n'est pas répétée. Elle est alors considérée comme étant une interaction sporadique. De même pour l'interaction  $\{\blacktriangleright\}$  du pas 10 au 12. Au pas 18, l'agent a créé les deux phénomènes attendus à savoir  $\langle \square \rangle$  et  $\langle \blacksquare \rangle$ . Mais ces phénomènes sont incomplets. Avec les différents tests, l'agent remplira correctement la tables d'usage des phénomènes  $\langle \square \rangle$  et  $\langle \blacksquare \rangle$  au pas 27. L'exploitation commence, avec des erreurs au pas 32. À partir de ce moment, l'agent se sert de ses croyances pour avancer, mais celui-ci se trompe encore car les tables d'usages ne sont pas correctement remplies. Ce n'est qu'à partir du pas 49 que les différents liens sont correctement établis et que l'exploitation peut se dérouler sans erreur.

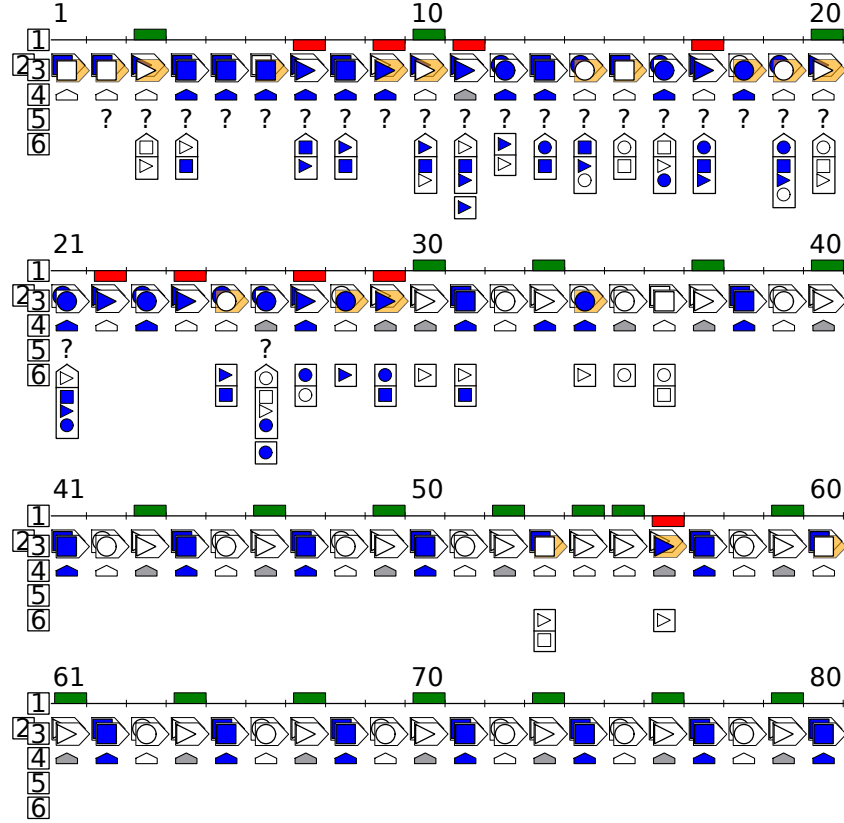
```

1 Function intendedMaxExpectedOutcomeValence(UsageTable believe)
2   si believe != NULL alors
3     pour chaque interactionList e of believe.getPostInteraction() faire
4       si e.isConfirm() alors
5         addAnticipation(e.getInteraction()
6           , e.getReward()
7           , e.getFactor()
8         )
9       fin
10    fin
11  sinon
12    pour chaque interactionList e of lastEnacted.getPostInteraction()
13      faire
14        si e.isPhenomena() alors
15          addAnticipation(e.getInteraction()
16            , e.getReward()
17            , e.getFactor()
18          )
19        fin
20    fin
21  retourner bestInteractionInAnticipation()

```

**Algorithme 3 :** Algorithme de sélection de la meilleur interaction.

**Système décisionnel n°2** Le second algorithme initialise les interactions comme étant persistantes et l'agent va alors apprendre au cours de son expérience les interactions sporadiques. La figure n°15 relate l'expérience de l'agent avec cet algorithme dans l'environnement du *string problem*. La figure n°16 montre l'évolution des états de croyance de l'agent.



**FIGURE 15.** 80 premières interactions avec l'algorithme d'apprentissage et d'exploitation en simultané. Du pas 1 au 21, l'agent possède des tables d'usages vides, il va alors remplir les tables d'usages en testant l'expérience la moins utilisée. Au pas 22, les tables d'usages permettent à l'agent de faire des suppositions et commencer l'exploitation de ses croyances : il pense que l'interaction  $\{\blacktriangleright\}$  amène au phénomène  $\{\triangleright\}$ . Au pas 25, l'interaction  $\{\blacktriangleright\}$  met à jour son état de croyance car il a *intended* l'interaction  $\{\bullet\}$  qui est une proposition de l'état de croyance  $\langle\triangleright\rangle$ . Comme l'interaction a échouée, l'état de croyance de l'interaction  $\{\blacktriangleright\}$  a évolué. Au pas 27, 30 et 35, l'agent apprend que les phénomènes  $\langle\bullet\rangle$ ,  $\langle\triangleright\rangle$  et  $\langle\circ\rangle$  sont en réalité des interactions sporadiques.

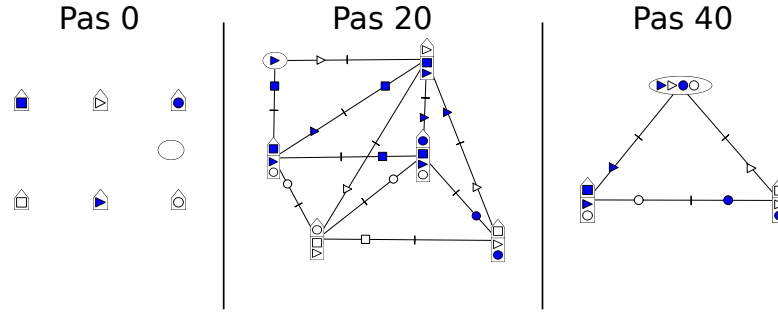


FIGURE 16. Évolution du graphe de Petri.

## 5 Un environnement pour le développement et l'étude d'agents développementaux

En parallèle de mes travaux sur la construction des connaissances par les agents, j'ai développé un environnement de travail destiné à accompagner ces travaux. Cet environnement, dont une interface est présentée figure n°17 permet de déployer différents types d'agents dans différents environnements, et de visualiser différents éléments au fil de l'exécution. Parmi ces éléments, on retrouve : les tables d'usage construites par l'agent, le graphe de Pétri déduit des tables d'usages, et enfin, les traces d'interaction produites par l'agent (interactions, évolution de la satisfaction, etc.).

Cet environnement, dont les fonctionnalités ont évoluées tout au long du stage, permet donc de supporter activement le développement et le débogage d'agents développementaux. Cela en permettant au concepteur d'exécuter différentes stratégies et de visualiser en temps réel les conséquences de ces stratégies sur le comportement des agents. De plus, l'outil permet de produire des visualisations graphiques explicites qui peuvent être exportées afin d'expliquer et d'illustrer les concepts mis en œuvre dans les agents. À titre d'exemple, l'ensemble des traces présentées dans ce rapport a été généré depuis l'environnement.

J'ai développé cet environnement dans l'optique de faciliter son utilisation et son extension par d'autres. Il est composé de 4 bibliothèques ainsi que d'un programme principal qui les lient entre elles.

La première bibliothèque donne l'implémentation des agents, des mécanismes d'apprentissage et d'exploitation des interactions. C'est cette bibliothèque qu'il faut enrichir si l'on souhaite étendre les mécanismes d'apprentissage ou expérimenter de nouvelles stratégies d'exploitation des interactions.

La seconde bibliothèque permet de réaliser une simulation d'un agent dans un environnement donné.

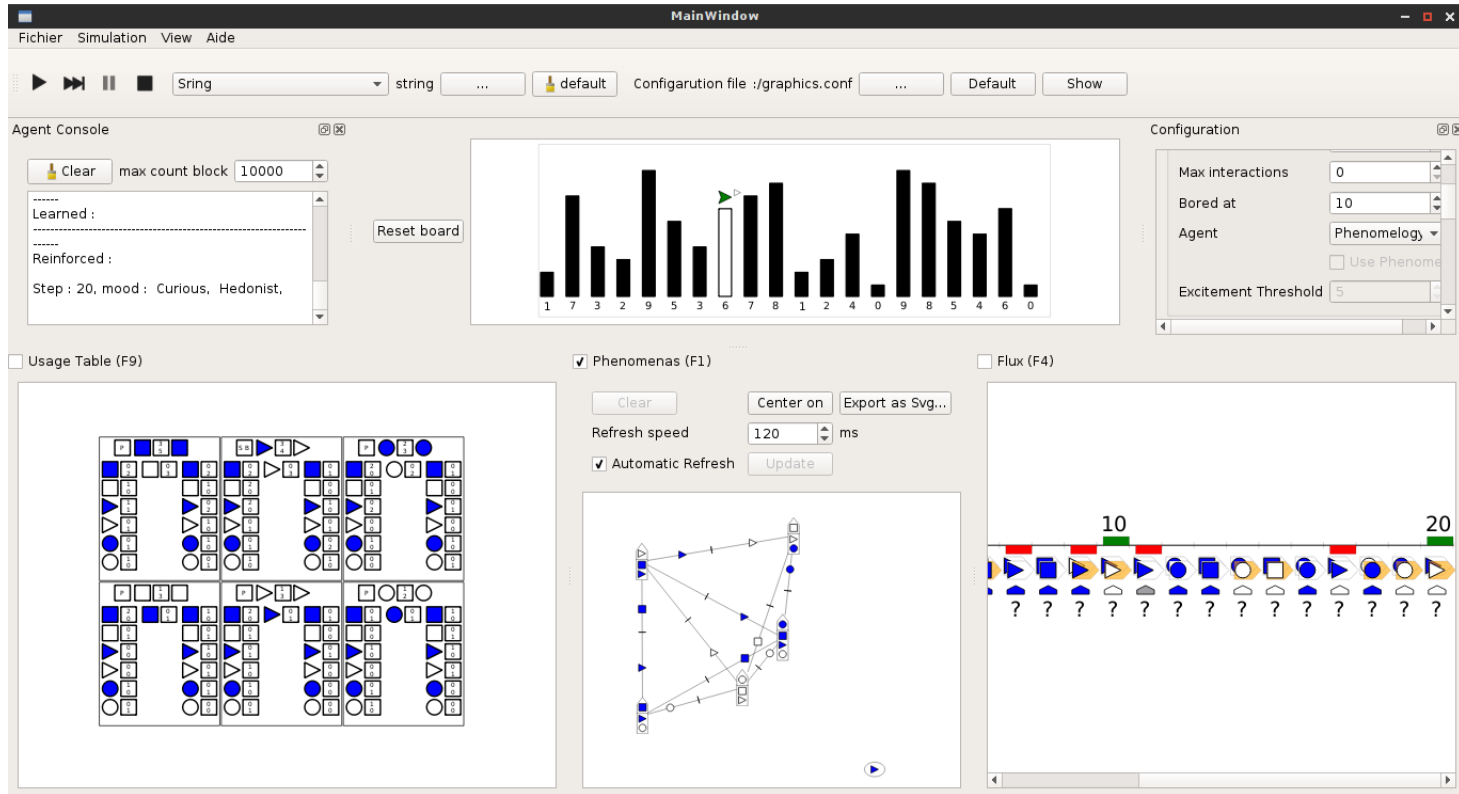
La troisième bibliothèque contient les outils requis pour la construction des visualisations graphiques des interactions et des tables d'usage.

La dernière librairie permet de lier l'environnement à un serveur de type KTBS<sup>7</sup>. Les KTBS sont des outils dédiés au stockage de traces. Cette librairie permet donc d'enregistrer toutes les traces d'exécution d'une expérimentation dans un serveur dédié pour pouvoir, par exemple, les réutiliser de façon asynchrone sans avoir à ré-exécuter les agents. Actuellement, cette fonctionnalité de réutilisation n'est pas implémentée, mais lorsqu'elle le sera, cela permettra aux utilisateurs de pouvoir visualiser et comparer des résultats d'expérimentations à tout moment sans avoir à reconfigurer l'ensemble de l'environnement pour chaque expérimentation.

Cet environnement a été intégralement développé en C++ et est disponible sous licence GPL à l'adresse suivante : <https://gforge.liris.cnrs.fr/anonscm/git/devlearnremiflo/devlearnremiflo.git>

---

7. KTBS : Kernel for Trace Base System dont les concepts sont disponibles à l'adresse : <https://kernel-for-trace-based-systems.readthedocs.org/en/latest/concepts/overview.html>



**FIGURE 17.** Dans cet exemple, un agent évolue dans le *string problem*. La partie inférieure propose plusieurs types de visualisation. Dans la partie supérieure droite se trouve la configuration de la simulation. L’affichage des logs de l’agent, apparaissent eux à gauche. On peut y voir les interactions activées, proposées, *intended*, *enacted*, apprises et renforcées et les humeurs de l’agent. Dans le menu, il est possible de sélectionner l’environnement avec son fichier de configuration. À droite, le fichier de configuration du graphisme.

## 6 Conclusion

L'approche développementale permet de construire un agent agnostique qui ne possède pas d'information sur son environnement. À travers ses interactions, l'agent va apprendre à interagir avec le monde pour satisfaire sa motivation interactionnelle. Ce rapport montre comment, à partir d'un flux d'interactions, un agent peut construire des connaissances liées aux régularités observées. L'agent représente des « choses » à travers ses interactions sans pour autant avoir un accès direct aux « choses ». Cette représentation soulève et laisse ouvertes de nombreuses questions comme : comment représenter des phénomènes avec des interactions composites, est-ce que les tables d'usage associées aux interactions composites donneront à l'agent la possibilité d'appréhender les environnements spatiaux impliquant la présence de plusieurs phénomènes à des endroits différents ?

Du côté de la motivation décisionnelle, nous avons détaillé deux algorithmes mettant en œuvre des formes d'excitation : la curiosité et l'hédonisme. D'autres motivations existent mais n'ont pas été implémentées :

- la simulation interne d'une interaction composite basée sur les connaissances de son environnement dans la perspective de trouver, voire de créer, une interaction plus satisfaisante ;
- la possibilité de se concentrer sur un objectif (représenté par une interaction primitive ou composite) pour déterminer s'il est atteignable ou non.

## Références

- And03. Michael L Anderson. Embodied cognition : A field guide. *Artificial intelligence*, 149(1) :91–130, 2003.
- Bro91. Rodney A Brooks. Intelligence without representation. *Artificial intelligence*, 47(1) :139–159, 1991.
- CKL94. Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the twelfth national conference on Artificial intelligence (AAAI 1994)*, pages 1023–1028, Menlo Park, CA, USA, 1994. AAAI Press.
- GA13. Olivier Georgeon and David Aha. The Radical Interactionism Conceptual Commitment. *Journal of Artificial General Intelligence*, 4(2) :31–36, December 2013.
- Gay14. Simon Gay. *Mécanismes d'apprentissage développemental et intrinsèquement motivés en intelligence artificielle : étude des mécanismes d'intégration de l'espace environnemental*. Thèse de doctorat en informatique, Université Lyon 1, December 2014.
- GC14. Olivier L Georgeon and Amélie Cordier. Inverting the interaction cycle to model embodied agents. *Procedia Computer Science*, 41 :243–248, 2014.
- GH13. Olivier Georgeon and Salima Hassas. Single agents can be constructivist too. *Constructivist Foundations*, 9(1) :40–42, November 2013.



- Gib77. JJ Gibson. *The theory of affordances*, In” *Perceiving, Acting and Knowing*”, Eds. RE Shaw and J. Bransford. Erlbaum, 1977.
- GMG12. Olivier Georgeon, James Marshall, and Simon Gay. Interactional Motivation in Artificial Systems : Between Extrinsic and Intrinsic Motivation. In *International Conference on Development and Learning (ICDL-Epirob)*, November 2012. Article court, communication affichée.
- GS12. Olivier Georgeon and Ilias Sakellariou. Designing Environment-Agnostic Agents. In Peter Vrancx Enda Howley and Matt Knudson, editors, *ALA2012, Adaptive Learning Agents workshop, at AAMAS2012, 11th International Conference on Autonomous Agents and Multiagent Systems*, pages 25–32, June 2012.
- Hus76. Edmund Husserl. *La crise des sciences européennes et la phénoménologie transcendantale*, page 262. Gallimard, 1976.
- MFF<sup>+</sup>97. Akira Murata, Luciano Fadiga, Leonardo Fogassi, Vittorio Gallese, Vassilis Raos, and Giacomo Rizzolatti. Object representation in the ventral premotor cortex (area f5) of the monkey. *Journal of neurophysiology*, 78(4) :2226–2230, 1997.
- O’R11. J.K. O’Regan. *Why Red Doesn’t Sound Like a Bell : Understanding the Feel of Consciousness*. Oxford University Press, USA, 2011.
- Pia59. J. Piaget. *The Construction of Reality in the Child*. The Basic classics in psychology. Basic Books, 1959.
- Ste04. Luc Steels. The autotelic principle. In *Embodied Artificial Intelligence*, pages 231–242. Springer, 2004.
- TNSW13. K Thórisson, Eric Nivel, Ricardo Sanz, and Pei Wang. Approaches and assumptions of self-programming in achieving artificial general intelligence. *Journal of Artificial General Intelligence*, 3(3) :1–10, 2013.
- VT. F Varela and E Thompson. et rosch e.(1991). the embodied mind : cognitive science and human experience.
- WF13. D.B. Webster and R.R. Fay. *The Mammalian Auditory Pathway : Neuroanatomy*. Springer Handbook of Auditory Research. Springer New York, 2013.