Écosystème
Un Framework pour la Simulation Visuelle Interactive Assistée

Thèse

Richard Drouin

Doctorat en Génie Électrique

Philosophiae Doctor (Ph.D.)

Québec, Canada

Résumé

La simulation visuelle interactive (SVI) permet d'analyser et de comprendre des systèmes complexes. Par contre, leur compréhension est limitée par la capacité humaine de traiter et d'analyser l'information véhiculée par la simulation afin de prendre de bonne décision concernant son déroulement futur. Le projet présenté dans cette thèse vise à aider la compréhension d'un scénario simulé par l'utilisation d'un Framework qui permet de simplifier la tâche d'analyse de l'utilisateur. En effet, le Framework effectue de façon autonome des tâches d'analyse et de traitement pour informer l'utilisateur du type d'interactions entre les entités du scénario simulé. Pour ce faire, le Framework permet tout d'abord de structurer les éléments de la simulation en s'inspirant du paradigme des écosystèmes. Ensuite, le Framework trouve et identifie les types d'interactions présentes dans le scénario. Finalement, le Framework informe l'utilisateur de façon claire et simple des comportements à noter. Comme les résultats obtenus le démontrent, le Framework permet de simplifier la tâche d'analyse d'un utilisateur de SVI en l'informant du type d'interactions présentes dans le scénario à l'étude.

Mots clés : Simulation visuelle interactive, logique floue, écosystème, interaction biologique, détection d'interaction, Framework, architecture logiciel, interaction, biomimétisme, système complexe, modélisation.

Table des matières

Liste des figures

Liste des équations

Liste des tableaux

"Any intelligent fool can make things bigger and more complex... It takes a touch of genius and a lot of courage to move in the opposite direction."

Albert Einstein (1879-1955)

Introduction

Mise en contexte

La simulation a toujours fait partie intégrante du monde moderne. D'une part, elle est principalement utile dans les secteurs d'activité utilisant le prototypage et les secteurs ayant recours à l'entraînement. L'utilisation de prototypes permet de simuler ce que sera le produit final. L'industrie de l'automobile, de l'aviation, du militaire et de la consommation sont des exemples de domaines ayant recours à l'utilisation de prototypes lors de la recherche et du développement de nouveaux produits. Les constructeurs automobiles utilisent les souffleries pour tester les nouveaux modèles d'automobiles afin de connaître les défauts de comportement dus à la friction de l'air ou pour tester l'aérodynamisme du véhicule. Pour sa part, le domaine de la consommation, utilise des prototypes de produits pour effectuer des études de marketing afin d'évaluer si le produit a sa place sur le marché. D'autre part, la simulation est aussi très efficace comme moyen d'entraînement. Le domaine militaire est probablement l'un des plus grands utilisateurs de la simulation. En effet, lors de l'entraînement des troupes, différents scénarios d'attaque et de défense sont simulés afin que les soldats soient adéquatement préparés lors d'un déploiement en milieu hostile. La fidélité des scénarios est la clé dans un contexte d'entraînement et ce, peu importe le domaine ciblé.

Avec la venue de l'ordinateur, la simulation a pris une autre voie, celle du numérique. L'ordinateur permet désormais de simuler différents phénomènes mathématiques et physiques de grande complexité. Par le fait même, l'envergure des simulations sur ordinateur[1] peut maintenant excéder tout ce qui était réalisé avec les méthodes conventionnelles de simulation qui consistaient à concevoir un prototype physique du produit ou du scénario à simuler. Tout ce qui était impossible à simuler il y a quelques années est maintenant possible. Le premier déploiement à grande échelle d'une simulation fut réalisé durant le projet Manhattan lors de la deuxième guerre mondiale. La simulation permettait de modéliser le processus de détonation nucléaire. À partir de ce point, le domaine de la simulation a évolué main dans la main avec l'avancement rapide des ordinateurs. La simulation fut dès lors utilisée dans de nombreux domaines comme: l'industrie, la fabrication de semi-conducteurs, la gestion de projet, l'ingénierie, le militaire, la logistique, le transport, les affaires et la santé [1]. Le Blue Brain Project [2] est un projet qui démontre la capacité actuelle de la simulation. Il a comme objectif principal de créer la première simulation d'un cerveau humain jusqu'au niveau moléculaire. Le modèle

[1] Le terme simulation sera désormais utilisé afin d'alléger le texte.

qui sera réalisé permettra de mieux comprendre le fonctionnement du cerveau humain. Donc, l'étendue des capacités et des domaines d'application a mené à la création de plusieurs sous domaines à la simulation.

Par exemple, la simulation visuelle interactive est un sous-domaine émergent de la simulation et permet d'exécuter un scénario en gardant un humain dans la boucle lors de la simulation du modèle. Par le fait même, une simulation visuelle interactive est un bon outil pour l'entraînement. Par exemple, un simulateur d'avion permet d'entraîner les apprentis pilotes en éliminant les dangers d'accident. En effet, différents scénarios catastrophes peuvent être simulés et permettre au pilote en devenir d'expérimenter les scénarios dans un contexte sécuritaire. La simulation visuelle interactive permet d'une part de simuler différents scénarios pour lesquels il est impossible d'évaluer à l'avance toutes les combinaisons d'entrées et de sorties et, d'autre part, elle permet à l'utilisateur d'interagir en temps réel avec le scénario. Elle peut s'avérer très utile dans des contextes complexes.

Le domaine de recherche sur les systèmes complexes peut bénéficier de l'utilisation de la simulation visuelle interactive. Un système complexe comporte plusieurs entités qui interagissent entre elles et qui génèrent un comportement global difficilement prévisible par l'observateur. De plus, le comportement d'un tel système est souvent non linéaire et ne peut résulter de la sommation du comportement de chaque entité. Donc, la simulation visuelle interactive permet l'exécution de scénarios impliquant des systèmes complexes durant laquelle un expert du domaine peut intervenir et influer sur l'orientation de la simulation. Par contre, une telle approche renferme plusieurs problèmes concernant la structuration de la vaste quantité d'information générée par la simulation, l'analyse en direct de cette dernière et sa représentation conceptuelle.

Cette thèse se concentre sur les problèmes soulevés au paragraphe précédent soit : structurer l'information, l'analyse en direct de celle-ci et sa représentation à l'utilisateur. Les travaux représentent donc une nouvelle branche de la simulation visuelle interactive de systèmes complexes (Figure 1). En effet, ils abordent les problèmes mentionnés d'un nouveau point de vue en proposant une nouvelle approche de simulation visuelle interactive basée sur les interactions entre les entités simulées. La prochaine section décrit plus en détails les problèmes que la nouvelle méthode suggérée tente de résoudre.

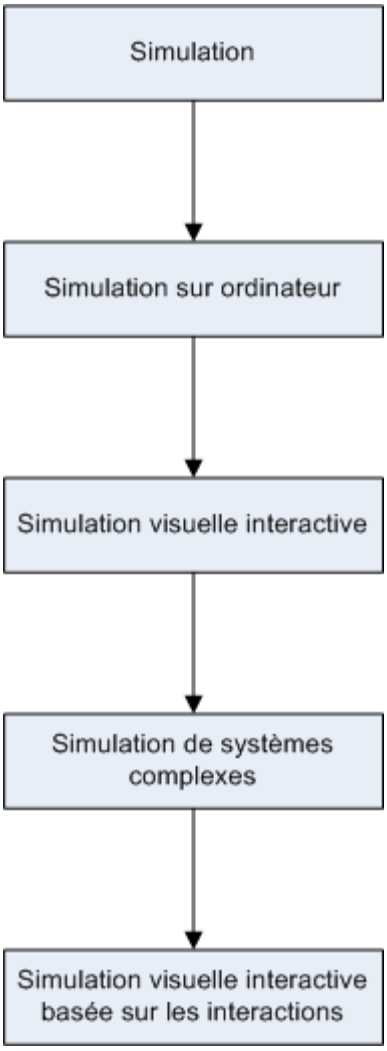


Figure 1 Origine de la simulation visuelle interactive basée sur les interactions

Problématique

Avec l'évolution constante de la puissance de calcul et des capacités graphiques de l'ordinateur, la simulation visuelle interactive de systèmes complexes est devenue un domaine de recherche et d'application très actif. En effet, avec l'impossibilité de prédire toutes les possibilités de comportement d'un système complexe, l'insertion d'un humain avec son savoir-faire dans le processus de simulation s'avère essentielle. Par contre, la gestion de l'information devient un problème, car cette dernière doit être traitée efficacement afin d'aider l'utilisateur dans sa tâche. En effet, une gestion efficiente de l'information est la clé du succès pour les utilisateurs de la simulation visuelle interactive. En consultant plus facilement l'information utile l'utilisateur pourra prendre des décisions plus adéquates et réfléchies concernant l'évolution des systèmes simulés. C'est

pour cette raison que les problèmes de gestion d'information en cours de simulation visuelle interactive doivent être abordés, notamment en ce qui concerne sa représentation à l'utilisateur et son intégration aux engins de simulation existant.

Premièrement, la représentation de l'information est une partie importante de la simulation visuelle interactive. En effet, elle est la base du modèle simulé. Ceci implique que l'information doit être organisée de manière adéquate et claire pour qu'elle soit facilement interprétée par l'utilisateur afin qu'il puisse prendre des décisions judicieuses concernant le futur de la simulation. Par le fait même, le choix de l'information pertinente et la façon dont elle sera représentée sont deux facteurs à ne pas négliger pour permettre une bonne intégration de l'information par l'utilisateur de simulation visuelle interactive.

D'une part, l'utilisateur doit avoir le plus d'information possible afin de prendre de bonnes décisions et de comprendre convenablement le déroulement du scénario. Par contre, le système ne doit pas simplement afficher toute l'information disponible, car l'utilisateur serait surchargé au plan cognitif. Ceci impliquerait que ses décisions puissent s'avérer inadéquates ou hâtives. En effet, afficher de l'information sans se soucier de la compréhension qu'un utilisateur peut en faire est une erreur. Les simulations visuelles interactives doivent seulement représenter ce qui est pertinent à l'apprentissage et utile à l'évolution de la simulation. À ce stade une question se pose : « Quelle information est pertinente et laquelle ne l'est pas? ». C'est ce à quoi doit répondre système de filtrage de l'information dans un contexte de simulation. Une approche de filtrage basée sur la détection des interactions entre les entités simulées sera proposée dans cette thèse.

D'autre part, l'information doit être représentée de façon claire, concise et ergonomique. En effet, une fois le filtrage de l'information effectué, cette dernière doit être transmise à l'utilisateur afin de lui permettre de comprendre de façon juste la simulation à des fins de décision. Donc, le format de l'affichage ne doit pas être mis de côté afin que l'expérience de simulation visuelle interactive soit profitable et enrichissante.

Deuxièmement, pour effectuer un filtrage adéquat de l'information, celle-ci doit être emmagasinée de façon efficace à l'intérieur du simulateur. Cette composante relative au stockage est transparente à l'utilisateur de la simulation visuelle interactive. Dû à la grande quantité d'information générée par le déroulement du scénario, l'engin de simulation est chargé de données de toutes sortes. On y retrouve un ensemble de variables qui ne sont pas toutes nécessaires à la compréhension de ce qui se produit dans le scénario. Par exemple, les états du simulateur ne sont pas pertinents lorsque l'on veut comprendre un phénomène simulé. Par ailleurs, les

attributs des entités simulées, pour leur part, sont d'une grande importance. Donc, structurer l'information sur la simulation de façon à ce qu'elle soit facilement accessible et utilisable est essentiel.
Les nombreux problèmes soulevés concernant l'organisation des données de simulation, le filtrage de ces données et leur représentation sont à la base des objectifs de cette thèse présentés dans la prochaine section.

Objectifs

Les problèmes discutés précédemment sur la gestion, la représentation et le filtrage des données soulèvent plusieurs défis de taille et doivent d'être solutionnés pour permettre une meilleure expérience pour l'utilisateur lors de l'exploitation de la simulation visuelle interactive. La présente section présente les objectifs de cette thèse qui visent à résoudre les problèmes illustrés précédemment et qui peuvent être décrits en trois mots soit : organiser, filtrer et informer (Figure 2). Un Framework (Figure 3) sera proposé afin de répondre aux besoins d'organisation, de filtrage et de synthèse d'information valable pour la simulation visuelle interactive. Il chevauchera les étapes d'exécution du modèle et de l'analyse de l'exécution ce qui implique qu'il fournira à l'utilisateur une analyse s'exécutant en arrière-plan en cours de simulation. Les étapes de la simulation sont couvertes en détail à la section « Simulation » du « Chapitre 1 : État de l'art ».

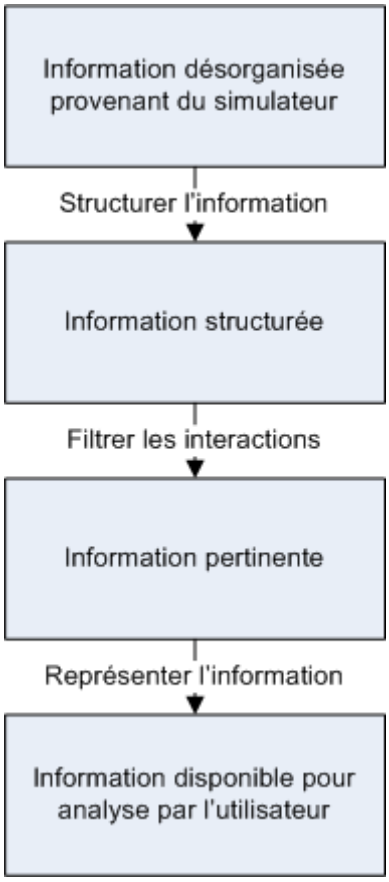


Figure 2 Objectifs (flèches) de la thèse

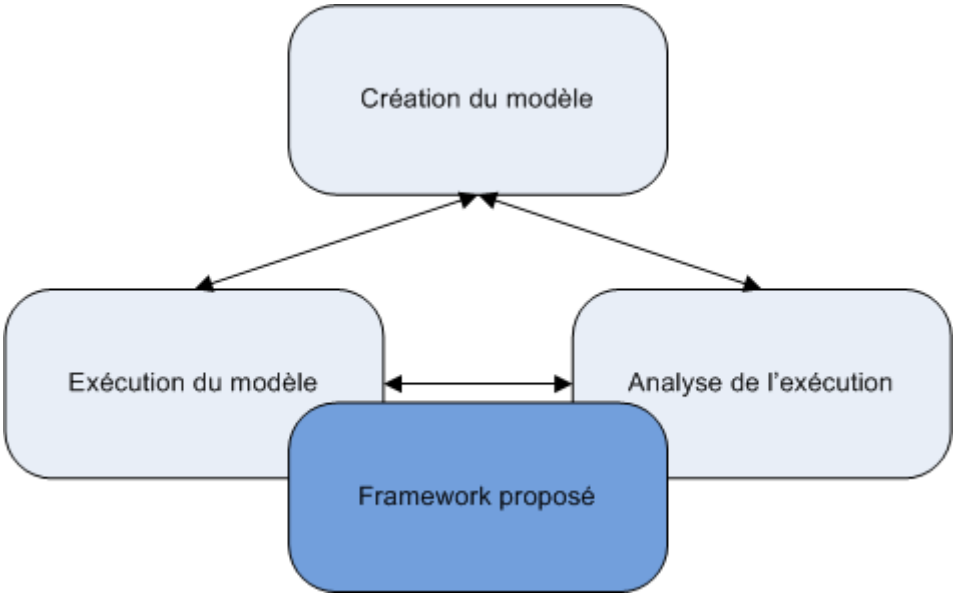


Figure 3 Situation du Framework proposé dans le cadre général de la simulation

Premièrement, l'organisation de l'information au niveau interne de l'engin de simulation est primordiale (« Structurer l'information » sur la Figure 2) pour qu'elle permette à l'utilisateur et au développeur de simulations de trouver aisément ce qu'ils cherchent. C'est donc pour cette raison que la façon dont l'information doit être structurée est abordée dans cette thèse et qu'une solution sera proposée par le Framework. Ce Framework permettra à un créateur de simulations de structurer l'information de façon simple et ordonnée. Cette organisation facilitera l'étape de filtrage suivante.

Deuxièmement, une fois l'information structurée convenablement, le développeur doit avoir la possibilité de manipuler les données à son aise soit en accédant directement à la donnée soit en la transformant en une forme plus utile par des algorithmes (« Filtrer l'information » sur la Figure 2). Ainsi, le Framework proposé permettra au développeur d'effectuer deux opérations : filtrages et traitement. De plus, des algorithmes permettant la détection des interactions entre les entités simulées seront offerts. Une interaction est détectée lorsque deux entités ont une influence réciproque [3]. L'information sur les interactions sera à la base du système de représentation qui est visé comme dernier objectif.

Finalement, informer l'utilisateur des changements pertinents des données en cours de simulation visuelle interactive est la dernière étape (« Représenter l'information » sur la Figure 2). Cette représentation de l'information est une part essentielle de la simulation. En effet, c'est le contact direct que l'utilisateur et l'expert ont avec le modèle simulé. L'information qui est présentée à l'utilisateur va donc décider du futur de la

simulation, car l'expert prendra des décisions en fonction de celui-ci. Donc, une base d'algorithmes pertinents pour permettre à l'utilisateur de bien suivre la simulation sera intégrée au Framework proposé.

En résumé, le Framework présenté dans cette thèse propose une solution aux problèmes d'organisation de l'information, de filtrage sélectif de cette dernière et offre la possibilité à l'utilisateur d'accéder aux données en plus de fournir des algorithmes permettant d'informer efficacement l'utilisateur en cours de simulation. L'atteinte de ces objectifs engendrera des contributions aux domaines de la simulation visuelle interactive qui seront présentées dans la section suivante.

Contributions

Cette thèse aborde un domaine quasiment inexploré qui vise à faciliter l'analyse d'un système complexe en simulation visuelle interactive par la détection des interactions entre les entités simulées. Par ailleurs, les contributions visées par cette thèse prennent la forme d'un Framework. Les deux contributions majeures sont au niveau de l'architecture du Framework proposé et au niveau des algorithmes de détection d'interactions et de classification en cours de simulation.

D'une part, l'architecture proposée permet de structurer l'information pour des scénarios de simulation des systèmes complexes. En effet, un paradigme basé sur les écosystèmes [4] est développé afin de créer un modèle cohérent [5] pour des scénarios de systèmes complexes. Ce paradigme permet de généraliser la structure de l'information pour un vaste éventail de scénarios où la compréhension visuelle est basée sur l'interaction entre les entités.

D'autre part, en plus d'offrir une architecture générale basée sur les interactions entre parties, le Framework automatise la détection des interactions et qualifie ces dernières selon l'influence que les entités ont entre elles. La détection de ces interactions se base sur l'analyse dans le temps des paramètres des entités composant le scénario. Pour sa part, la classification se base sur le paradigme des interactions biologiques [6]. L'analyse temporelle des paramètres permet de détecter les interactions et de les classifier.

En résumé, le terme simulation visuelle interactive basé sur les interactions entre les entités simulées est défini pour la première fois. L'architecture proposée permet d'intégrer un scénario simulé sur un engin de simulation donné à une analyse permettant de détecter les interactions. Il sera possible grâce aux travaux de cette thèse d'informer l'utilisateur du type d'interactions présentes dans le scénario ce qui, à notre connaissance, n'a jamais été proposé auparavant.

Structure de la thèse

Le « Chapitre 1 : État de l'art » présente les domaines qui ont permis de définir le projet de recherche. Les domaines de la simulation, des systèmes complexes, de la modélisation et de la détection d'interactions lors d'une simulation seront abordés.

Le « Chapitre 2 : Modèle conceptuel et architecture » se concentre sur le modèle conceptuel qui a été mis en place dans le cadre de la thèse. De plus, le chapitre abordera la façon dont ce modèle est implanté à l'intérieur du Framework développé dans le cadre de la thèse.

Le « Chapitre 3 : Détection d'interactions » présente les deux types d'algorithmes fournis avec le Framework. D'une part, les calculs préliminaires à la détection d'interactions sont présentés. D'autre part, les algorithmes de détection et d'indentification des interactions sont détaillés.

Le « Chapitre 4 : Scénarios, résultats et discussion » aborde trois aspects. Tout d'abord, il présente les scénarios qui sont utilisés dans le cadre de la thèse et, ensuite, les résultats présentés et analysés afin de valider l'approche proposée.

Chapitre 1 : État de l'art

Ce chapitre présente une revue des travaux sur lesquels s'appuie cette thèse. Comme le domaine abordé est quasiment inexistant dans la littérature, l'état de l'art couvre les concepts qui seront utilisés dans la thèse ou les concepts qui ont été étudiés et écartés pour différentes raisons. De plus, les domaines influencés par les travaux seront discutés. Premièrement, le domaine de la simulation sera couvert. L'explication du concept, les avantages et les inconvénients de ce dernier ainsi que quelques exemples seront abordés afin de bien faire comprendre au lecteur le domaine de la simulation. Deuxièmement, le domaine des systèmes complexes en lien avec la simulation sera abordé. Troisièmement, les travaux sur les modèles conceptuels seront présentés. Finalement, des techniques de détection d'interactions entre entités de simulation seront couvertes.

Simulation

> « La simulation est la discipline qui consiste à la création d'un modèle d'un système actuel ou d'un système physique théorique, à l'exécution de ce dernier sur un ordinateur et à l'analyse des données en sortie de cette exécution. Par le fait même, la simulation inclut le principe d'apprentissage par essai. Donc, l'utilisation de la simulation est une activité aussi naturelle que le jeu de rôle l'est à l'enfant [7]. »

La simulation est un processus composé des trois étapes présentées à la Figure 4.

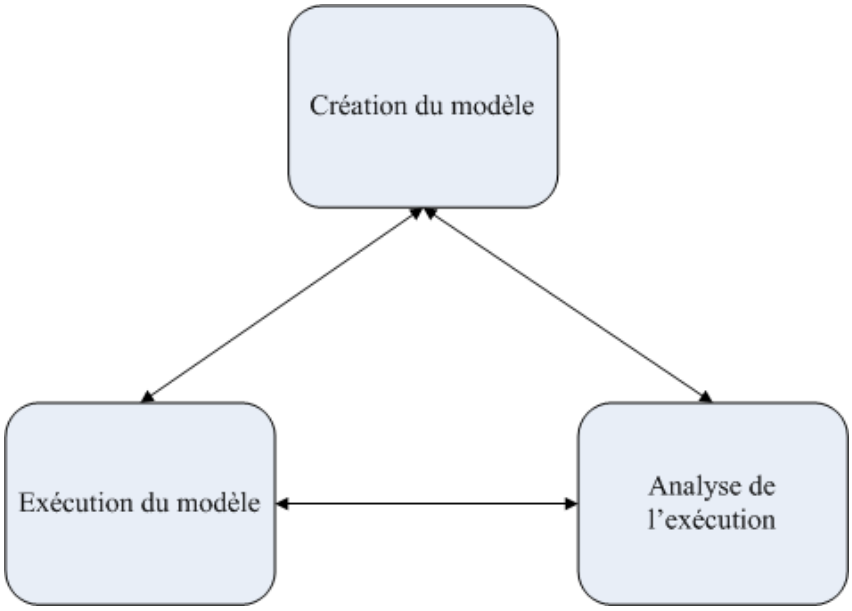


Figure 4 Étapes de la simulation [7]

La première étape est la création du modèle à partir d'un système réel ou théorique. La création du modèle se fait en trois parties qui sont représentées à la Figure 5 soit : l'observation, la construction du modèle conceptuel et l'implantation du modèle.

La première partie de la création du modèle consiste à l'observation du système réel et à l'acquisition de données sur ce système. Ces données peuvent être de forme symbolique ou numérique. D'une part, l'observation permet de mieux comprendre le système et les interactions entre les différentes composantes impliquées. Par contre, l'observateur devrait, dans le domaine du possible, s'informer auprès d'experts sur le système à évaluer. En effet, ceci lui permettra de comprendre les subtilités et de ne pas passer outre à certains comportements. D'autre part, l'acquisition de données permettra de valider le modèle qui sera réalisé. Les données recueillies peuvent être quantitatives ou qualitatives. Les données quantitatives sont obtenues grâce à l'utilisation de capteurs comme des capteurs de température ou de pression. Les données qualitatives sont obtenues par le biais d'entrevues faites auprès des experts du système. Les données quantitatives et qualitatives permettent donc de créer une base de connaissances approfondie sur le système qui sera modélisé.

La deuxième partie consiste à la création d'un modèle conceptuel avec l'information recueillie à la première étape. Lors de ce processus, plusieurs hypothèses sont formulées. En effet, certaines hypothèses sont faites concernant les composants et la structure du système ainsi que les paramètres d'entrée. Bien entendu, une validation conceptuelle est requise afin de s'assurer de la conformité du système conceptuel par rapport au système réel. Lorsque le modèle conceptuel est réalisé, l'implantation du système peut commencer.

La troisième partie consiste à implanter le système opérationnel. Dans la plupart des cas, cette étape est réalisée par l'intégration des hypothèses du modèle conceptuel à un engin de simulation. Bien entendu, la création d'un modèle n'est pas un processus linéaire à trois étapes. En effet, le concepteur sera appelé à revisiter les étapes précédentes lors de la conception, la vérification et la validation du modèle tel que présenté à la Figure 5.

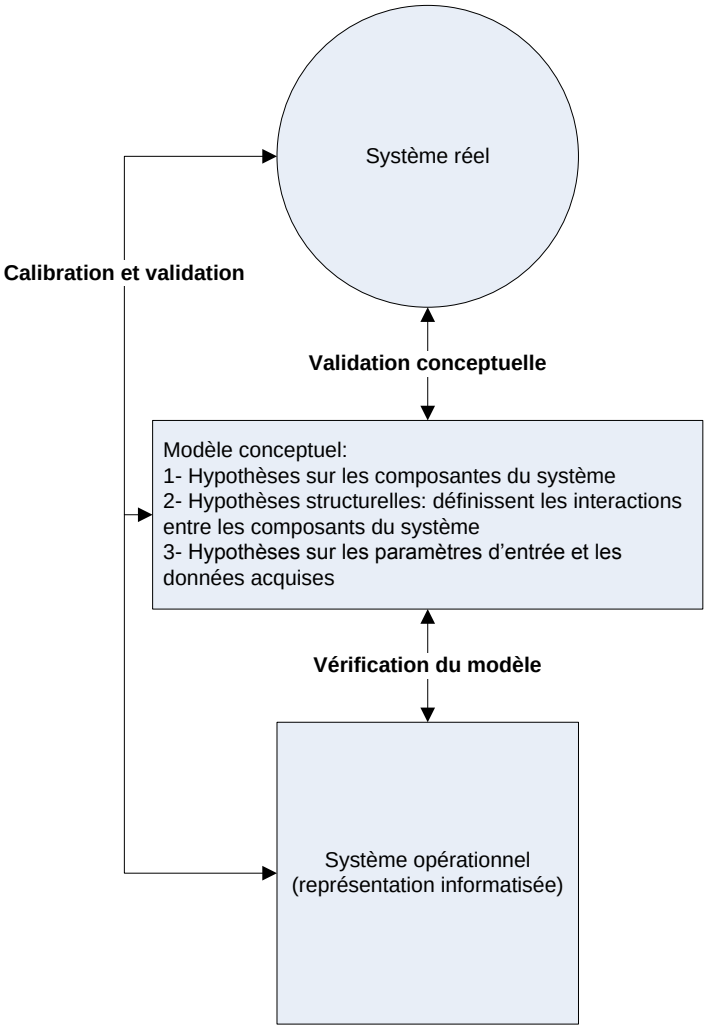


Figure 5 Construction, vérification et validation d'un modèle [1]

La deuxième étape du processus de simulation est l'exécution du modèle réalisé. Cette étape permet de raffiner le modèle et/ou d'obtenir des données sur le système final. D'une part, l'exécution permet d'obtenir des résultats partiels qui permettent de valider les parties de système réalisé jusqu'à présent. D'autre part, l'exécution permet d'obtenir les résultats du système final pour ensuite les analyser afin de mieux le comprendre.

La troisième étape est l'analyse des données fournies par le système modélisé. De nombreuses techniques d'analyse sont disponibles :

- Analyse des combinaisons entrées/sorties;
- Visualisation des données;
- Design expérimental [8];

- Technique « *response surface* » [9];
- Vérification et validation [10].

Chaque étape du processus de simulation a été présentée séparément, mais il faut noter que la création, l'exécution et l'analyse du modèle font partie d'un processus itératif et incrémental [11]. C'est-à-dire que chacune des phases de développement est divisée en itérations. Une itération est une boucle complète de développement résultant en un sous-produit de la solution finale. À chaque itération, le système offre de plus en plus de stabilité et de fonctionnalité jusqu'à la version finale [12].

Maintenant que les étapes du processus de simulation ont été présentées, il convient de poser la question : pour quels contextes la simulation est-elle le bon outil qui va permettre une meilleure compréhension d'un système existant ou théorique?

Quand la simulation est elle le bon outil?

> « La simulation est l'un des outils les plus puissants disponibles pour le design et l'opération de procédés et systèmes complexes. Elle permet d'étudier, d'analyser et d'évaluer des situations qu'il ne serait pas possible d'observer autrement. Dans un monde en constante compétition, la simulation est devenue une méthodologie indispensable pour résoudre des problèmes en ingénierie, en design ou en gestion. [13] »

Malgré le fait que la simulation soit un excellent outil, les circonstances et les contextes pour lesquels cet outil est approprié doivent être discutés. La simulation peut être utilisée avec pertinence dans les cas suivants [1] :

1. La simulation permet l'étude et l'expérimentation des interactions internes d'un système complexe ou d'un sous-système à l'intérieur d'un système complexe;
2. Les changements informationnels, organisationnels et environnementaux peuvent être simulés et les effets de ces altérations sur le comportement du modèle peuvent être observés;
3. Les connaissances acquises lors de la création du modèle peuvent suggérer des améliorations au système sous investigation;
4. Changer les valeurs d'entrée et observer les sorties résultantes afin d'offrir un aperçu des variables les plus importantes et comment ces variables interagissent à l'intérieur d'un système à l'étude;
5. La simulation peut servir comme outil pédagogique;
6. La simulation peut être utilisée pour expérimenter de nouveaux designs avant l'implantation concrète de façon à préparer l'utilisateur aux scénarios susceptibles de se produire;

7. La simulation peut être utilisée pour vérifier des solutions analytiques;
8. Simuler les capacités d'une machine peut aider à déterminer ses prérequis;
9. Le design d'un modèle à des fins pédagogique permet l'apprentissage à faible coût et élimine le besoin d'un instructeur;
10. Les animations permettent d'observer un système simulé en opération;
11. Les systèmes modernes sont d'une grande complexité, alors les interactions internes ne peuvent souvent être observées que par simulation.

Malgré les nombreuses circonstances où la simulation est un outil très intéressant, dans certains contextes il n'est pas conseillé de l'utiliser.

Quand la simulation n'est-elle pas le bon outil?

Plusieurs situations sont propices à l'utilisation de la simulation comme le démontre la section « Quand la simulation est elle le bon outil? ». Par contre, il est essentiel de connaître les situations où la simulation n'est pas le bon outil. Cette section est basée sur l'article de Banks[14] qui présente dix règles décrivant les situations où la simulation n'est pas appropriée. Voici les règles qui présentent les cas qui doivent utiliser une autre technique que la simulation pour être résolus :

1. Le problème peut être résolu avec le bon sens. L'exemple donné est celui des stations de service de soutien téléphonique où les appels arrivent aléatoirement à un taux de 100 par heure et sont traités à un taux de 12 par heure pour chaque employé. Pour déterminer le nombre minimal d'employés nécessaire pour répondre à la tâche, la simulation n'est pas nécessaire. En effet, il suffit de calculer 100/12 = 8.33 ce qui indique qu'un minimum de 9 employés est nécessaire afin d'accomplir la tâche. L'utilisation de la simulation dans ce cas aurait été beaucoup plus coûteuse.
2. Le problème peut-être résolu analytiquement. Par exemple, sous certaines conditions, le temps d'attente moyen pour un client dans l'exemple présenté dans la règle 1, peut-être calculé en utilisant les courbes développées par Hillier et Lieberman [15]. Ce qui est certainement plus rapide que de créer une simulation pour calculer la réponse.
3. Il est plus facile de changer ou d'effectuer les expérimentations directement sur le système réel. Par le passé nous avons vu certains cas où la création du modèle à pris plus de temps et coûté plus cher que l'expérimentation simple et directe. Prenons l'exemple d'un modèle qui a été créé pour simuler le service au volant d'une chaîne de restauration rapide afin de tester les conséquences sur le temps de service de l'ajout d'un seconde fenêtre de commandes. Le modèle a pris plusieurs semaines à

réaliser. Au même moment, un compétiteur a expérimenté la situation en simulant une seconde personne avec un terminal distant et un outil de communication sur le long de la ligne de commande. L'étude a été réalisée en quelques jours. Donc, si le problème implique un système existant qui peut être perturbé sans conséquences, il est préférable de vérifier si l'approche d'expérimentation directe peut répondre à la question. De plus, l'approche directe permet d'éviter les questions concernant la validité du modèle créé.

4. La simulation coûte plus cher que la mise au point d'un nouveau système. Voici les facteurs à prendre en compte lorsqu'il est temps de calculer les coûts d'un projet de simulation :
    - La planification du projet, la définition du problème et la documentation;
    - Le développement du modèle et les essais;
    - La collecte de données, la révision et le formatage;
    - La validation du modèle;
    - L'expérimentation et l'analyse;
    - Les améliorations possibles du modèle;
    - La documentation du projet et sa présentation.
5. Les ressources ne sont pas disponibles pour réaliser le projet de simulation. Pour compléter un projet de simulation les éléments suivants doivent être disponibles :
    - Des experts du domaine;
    - Des logiciels et les ordinateurs;
    - Des ressources financières.
6. Il n'y a pas assez de temps pour que les résultats du modèle soient utiles. Ceci se produit le plus souvent pour trois raisons :
    - Le temps alloué au projet est trop court;
    - Le développement du modèle et les tests prennent trop de temps;
    - La fenêtre de développement est trop étroite.
7. Les données ne sont pas disponibles. En effet, une des tâches critiques est de déterminer si les données requises pour la réalisation du projet sont disponibles. Si ce n'est pas le cas, comment est-il possible de les obtenir? Dans certains cas, il est impossible de les obtenir, car elles ne sont pas disponibles ou il serait trop coûteux de les obtenir.
8. La vérification ou la validation du modèle n'est pas possible. Ceci se produit lorsqu'un des trois éléments principaux est manquant : des experts, des données et du temps.

9. Les clients ont des attentes déraisonnables. Dans certains cas, ils attendent des résultats dans un laps de temps trop court ou ils surestiment la puissance de la simulation.
10. Le comportement du système est trop complexe à modéliser ou peut ne pas être défini correctement. C'est souvent le cas lorsque le comportement humain fait partie du système à simuler.

Avantages et inconvénients de la simulation

Lorsqu'il est opportun de réaliser la simulation d'un système, il en ressort de nombreux avantages ainsi que certains désavantages qui seront abordés dans cette section.

La simulation tend à reproduire les systèmes réels et c'est ce qui la rend attrayante comme technique d'analyse. En effet, il est possible d'apporter des modifications à un système existant ou de créer un nouveau système et de regarder comment ce dernier se comporte sous certaines contraintes. Donc, la simulation renferme plusieurs avantages, mais aussi certains inconvénients qui seront abordés en se basant sur la liste offerte par Pegen, Shannon et Sadowsky [16].

Les avantages de la simulation sont nombreux :

1. Il est possible de créer de nouvelles politiques, procédures d'utilisation, règles de décision, de nouveaux flots d'information et des procédures organisationnelles sans avoir à interrompre les opérations du système réel;
2. Il est possible de tester de nouveaux designs de matériel, des topologies physiques, des systèmes de transport etc. sans avoir à assigner des ressources à la collecte d'information;
3. Des hypothèses sur le pourquoi et le comment de certains phénomènes peuvent être validées;
4. L'échelle de temps peut-être augmentée ou diminuée pour le système sous investigation;
5. Des connaissances peuvent-être obtenues concernant les variables qui interagissent entre elles;
6. Des connaissances peuvent être obtenues concernant l'importance des variables et les performances du système;
7. Les goulots d'étranglement peuvent être découverts;
8. Uns simulation peut permettre de comprendre objectivement comment un système opère;
9. Des questions de type « *what-if* » peuvent trouver réponse.

Malgré les nombreux avantages, certains inconvénients associés à l'utilisation de la simulation peuvent être relevés. En voici les principaux :

1. La simulation est un art qui peut être maîtrisé avec le temps et l'expérience et son apprentissage est exigeant;
2. Les résultats de la simulation peuvent être difficiles à interpréter. En effet, la plupart des sorties d'une simulation sont essentiellement des valeurs aléatoires (elles sont pour la plupart du temps basées sur des entrées aléatoires), alors il peut-être difficile de distinguer si une observation est le résultat du système en évolution ou si elle est aléatoire;
3. La modélisation et l'analyse d'une simulation peuvent être coûteuses en temps et en argent;
4. La simulation peut être utilisée dans certains cas pour résoudre des problèmes où une solution analytique est possible (discuté dans la section « Quand la simulation est elle le bon outil? »).

Les conséquences associées aux quatre inconvénients peuvent être amoindries par les facteurs suivants :

1. Plusieurs d'engins de simulation existent sur le marché. Ces engins n'ont besoin que d'entrées pour opérer et sont raisonnablement facile à utiliser;
2. Le créateurs d'engins de simulation ont développé plusieurs fonctionnalités d'analyse de sorties de systèmes;
3. Les solutions analytiques sont pratiquement incapables de résoudre les problèmes complexes actuels. La simulation, même imparfaite, apporte quand même des éléments de solution utiles.

Domaines d'application de la simulation

La simulation offre la chance d'explorer des possibilités jusqu'à présent jugées impossibles à atteindre. Un survol de la conférence *Winter Conference Simulation (WCS)* [17] donne une idée de la vaste étendue des domaines d'application de la simulation comme en fait foi l'extrait suivant :

> *«* The Winter Simulation Conference (WSC) is the premier international forum for disseminating recent advances in the field of system simulation, with the principal focus being discrete-event simulation and combined discrete-continuous simulation. In addition to a technical program of unsurpassed scope and quality, WSC provides the central meeting place for simulation practitioners, researchers, and vendors working in all disciplines and in the industrial, governmental, military, and academic sectors.[17] *»*

La WSC est sponsorisée par de grandes associations reconnues pour leur contribution dans leur domaine. En voici la liste :

- American Statistical Association (ASA) [18];

- Association for Computing Machinery: Special Interest Group on Simulation (ACM/SIGSIM) [19];
- Institute of Electrical and Electronics Engineers: Systems, Man, and Cybernetics Society (IEEE/SMC) [20];
- Institute for Operations Research and the Management Sciences: Simulation Society (INFORMS-SIM) [21];
- Institute of Industrial Engineers (IIE) [22];
- National Institute of Standards and Technology (NIST) [23];
- The Society for Modeling and Simulation International (SCS) [24].

Le programme de la WCS de 2009 présente l'éventail de domaines touchés par la simulation. En voici la liste [25] :

1. Énergie;
2. Modélisation et analyse de production de semi-conducteurs;
3. Applications militaires;
4. Santé;
5. Application de production;
6. Logistique, transport et distribution;
7. Sécurité;
8. Modélisation de processus d'entreprise.

Depuis les débuts des années 1990 la conférence a dû s'adapter afin de rester à jour avec les développements rapides du domaine de la simulation. De nouveaux domaines d'application y sont ajoutés fréquemment.

Simulation visuelle interactive

Comme la section « Quand la simulation est elle le bon outil? » le montre, la simulation est un outil extrêmement performant et adapté à plusieurs domaines. La raison est très simple : les bénéfices engendrés par l'utilisation de la simulation sont énormes comparativement aux inconvénients possibles (voir la section « Avantages et inconvénients de la simulation »). C'est grâce à nombreux avantages que le domaine de la simulation s'est développé au point où de nombreuses entreprises optent pour cette méthode lors de développement de nouveaux produits. Un type de simulation particulièrement utilisé est la simulation visuelle

interactive (VIS). Cette section portera sur la VIS en décrivant le concept derrière cette méthodologie tout en citant les avantages qui s'y attachent.

La VIS a été conçue par Hurrion en 1976 lors de la réalisation de sa thèse de doctorat. Elle peut être définie comme suit:

> « Visual Interactive Simulation (VIS) is a type of simulation in which the user interacts with all phases of the simulation through a visual interface. The simulation user's ability to observe and interact with a running simulation through graphical interfaces is a fundamentally different way to involve a decision maker in all aspects of the simulation development and design [26]. »

La Figure 6 illustre bien le principe derrière la VIS.

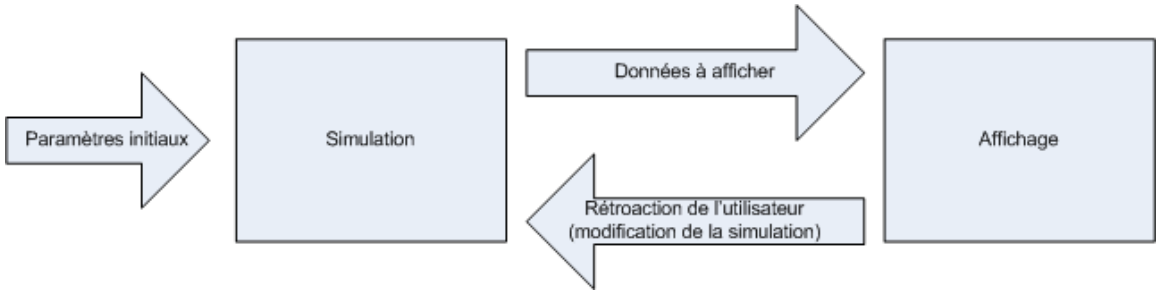


Figure 6 Concept de simulation interactive visuelle

Premièrement, les paramètres initiaux de la simulation sont choisis. C'est-à-dire que les entités de la simulation sont mises en place et que leurs propriétés sont initialisées. Par la suite, la simulation est lancée. Au même moment, différentes valeurs associées à la simulation sont affichées à l'utilisateur. Ceci lui permet d'analyser ce qui se passe en temps-réel dans la simulation. Finalement, l'utilisateur peut prendre des décisions concernant le futur de la simulation en interagissant avec cette dernière. Ceci permet d'inclure l'utilisateur dans le processus de simulation comme le résume Hurrion :

> « By watching a simulation model progress through time and having the ability to interact with it, the user can improve his analysis and understanding of the original problem situation.... The decision maker can watch the progress of the model and can himself contribute to the validation of the model. Hence, he will have more confidence in its use, because of his participation [27]. »

D'autres termes sont employés pour désigner ce type de simulation. La division de *Modeling and Simulation (M&S) United States Departement of Defense (DoD)* le qualifie d'approche *Human-in-the-Loop (HITL)*.

> « A model that requires human interaction [28]. »

Ceci n'implique pas nécessairement que l'interaction est le résultat de l'analyse visuelle, mais, dans la plupart des cas, c'est ce qui se produit. En effet, dès 1986, l'animation graphique est devenue si importante que chaque engin de simulation se doit d'offrir cette possibilité [29].

Ce type de simulation peut aussi être appelé *Interactive Real-Time Simulation*.

> « One can model only the computing system while leaving real components and any human participants as they are [30]. »

Une fois de plus, il n'est pas explicitement suggéré que la sortie de la simulation est de type visuel. Par contre, c'est dans la plupart du temps le cas.

La VIS est donc utilisée dans plusieurs domaines et particulièrement dans le domaine de grande complexité où la contribution de l'humain est nécessaire.

Systèmes complexes

Avec l'avancement de la science, l'être humain fait de plus en plus face à des problèmes de grande complexité. Sa fascination à comprendre ce qui l'entoure comme les écosystèmes, le système immunitaire humain ou l'économie, et les origines de certains phénomènes le poussent à poursuivre des recherches sur des systèmes de plus en plus complexes. Cette section présente un survol des notions pertinentes aux systèmes complexes. Tout d'abord la définition et les caractéristiques de tels systèmes seront présentées. Ensuite, l'historique de cette discipline sera abordé. Finalement, le rapport entre le domaine des systèmes complexes et la simulation sera traité. Cette section s'inspire du travail de Bernard Pavard et Julie Dugdale [31].

Définition

Voici quelques définitions du terme système complexe :

> « A complex system is a system composed of interconnected parts, that as a whole, exhibit one or more properties (behavior among the possible properties) not obvious from the properties of the individual parts [32]. »

> « A complex system is a network of heterogeneous components that interact nonlinearly, to give rise to emergent behavior [33]. »

> « Complex systems are systems where the collective behavior of their parts entails emergence of properties that can hardly, if not at all, be inferred from properties of the parts [34]. »

Toutes les définitions précédentes tentent d'expliquer la nature de base d'un système complexe. C'est-à-dire qu'un tel système est formé de parties qui interagissent de façon non-linéaire. Un comportement complexe est imprévisible ou adopte des comportements émergents. Il est important de faire la distinction entre un système complexe et un système compliqué. Le système compliqué est composé de plusieurs parties qui fonctionnent de façon autonome. Un avion et un ordinateur sont des exemples de systèmes compliqués. Par contre, le comportement global du système est prévisible. Le système complexe est composé pour sa part de parties qui interagissent de façon non-linéaire avec leur environnement et les autres composants, ce qui crée de l'auto-organisation et des comportements émergents. Les écosystèmes et la bourse font partie des systèmes complexes avec une forte composante non-linéaire et imprévisible. À partir de la définition, il est maintenant possible d'identifier certaines propriétés qui caractérisent tous les systèmes complexes.

Propriétés des systèmes complexes

Les systèmes complexes possèdent de quatre propriétés principales : non-déterminisme, décomposition fonctionnelle limitée, caractère distribué et émergence de comportements.

La première propriété est qu'un système complexe est non-déterministe. Ceci signifie qu'il est impossible d'anticiper précisément son comportement même si les fonctionnalités de ses constituants sont connues.

> *« A system in which the output cannot be predicted because there are multiple possible outcomes for each input* [35]. *»*

La physique quantique renferme quelques exemples de système non-déterministes comme la désintégration radioactive [36] qui est complétement aléatoire lorsque prise au niveau de l'atome.

La seconde propriété est qu'un système complexe se prête à une décomposition fonctionnelle limitée et adopte une structure dynamique. Par le fait même, il est très difficile sinon impossible de décomposer un tel système en sous-parties fonctionnelles. En effet, l'interaction permanente entre le système et son environnement ou entre les parties composant le système et sa capacité de s'auto-organiser lui permettent de se restructurer de lui-même. Un exemple concret est la réorganisation de la division des tâches dans une société d'insectes.

> « A key feature of the division of labor in insect colonies is its plasticity. Colonies respond to changing internal and exteral conditions by adjusting the ratios of individual workers engaged in the various tasks. This is accomplished in large part via the behavioral flexibility of the individual workers themselves. Worker behavioral flexibility contributes to the reproductive success of a

> colony by enabling it to continue to grow, develop, and ultimately produce a new generation of reproductive males and females despite changing colony conditions [37]. »

Dans un tel cas, différentes tâches sont effectuées simultanément par des individus spécialisés. Par contre, la division de différentes tâches entre les groupes de travailleurs n'est pas rigide. En effet, certains travailleurs peuvent effectuer plus d'une tâche afin d'ajouter de la flexibilité à la colonie et une meilleure capacité de reproduction. Différents facteurs peuvent causer le changement de rôle au sein de la colonie : la disponibilité de la nourriture, la prédation ou les changements climatiques.

La troisième propriété est qu'un système complexe est distribué. En effet, un tel système complexe possède plusieurs caractéristiques propres aux systèmes distribués en ce qui a trait aux connexions entre les éléments qui le composent. Certaines fonctions ne peuvent pas être localisées précisément dans le système. De plus, les éléments existant à l'intérieur du système complexe sont non-linéaires et contiennent des rétroactions positives et négatives.

La quatrième propriété est qu'un système complexe inclut des fonctionnalités émergentes qui ne sont pas révélées par l'analyse des composants individuels du système. L'émergence est le processus par lequel de nouvelles structures, patrons ou propriétés cohérents sont dérivés à l'intérieur d'un système complexe. Cette émergence survient lorsque de nouveaux patrons d'interaction s'installent avec le temps. Un exemple classique d'émergence est le monticule créé par les termites (Figure 7). Dans ce cas précis la citation d'Aristote prend tout son sens :

> « The whole is something over and above its parts, and not just the sum of them all. »

La création du monticule par les termites est le résultat d'interactions simples entre ces derniers sans que ce soit l'objectif visé au préalable.

Les propriétés énoncées précédemment représentent les prérequis pour qu'un système soit caractérisé comme étant complexe. L'origine de la notion de systèmes complexes sera abordée dans la section suivante.

Figure 7 Monticule créé par des termites (Photo par : Brian Voon Yee Yap)

Historique des systèmes complexes

La notion de système complexe à vu le jour au début du 20ième siècle. En effet, c'est lorsque Henri Poincaré a travaillé sur des équations permettant de prédire la trajectoire de planètes qu'il a fait la découverte de la notion de systèmes complexes. Il a montré qu'il était mathématiquement impossible de trouver une solution exacte

aux équations de trajectoire des planètes même si le système ne contenait que trois corps interagissant de façon non-linéaire. À ce moment il a fait part de sa trouvaille à la communauté scientifique :

> « Even a completely causal system (a system where the behavioural rules are perfectly known) could have indeterminate behaviour [31]. »

En d'autres termes il a démontré comment un système apparemment simple pouvait produire un comportement complexe et potentiellement imprévisible.

Le point qui a intrigué le plus Poincaré est le suivant : comment un système déterministe du point de vue des ses fonctionnalités peut-il avoir un comportement imprévisible? À cette époque, l'absence d'ordinateurs puissants empêchait de trouver des solutions à des problèmes n'ayant pas de solution exacte. Il était donc pratiquement impossible d'explorer le nouveau domaine des systèmes complexes.

Simulation et systèmes complexes

La nature même d'un système complexe rend impossible la prédiction de son comportement exact. En effet, même si le comportement de chacun des composants du système est bien connu, il est tout de même impossible de prédire le comportement global du système. Les approches statistique ou mathématique ne sont pas appropriées, car les équations décrivant les non-linéarités sont difficiles sinon impossible à établir. La seule façon d'explorer le domaine non-linéaire est par l'utilisation de la simulation.

Modèle conceptuel de données de simulation

L'objectif du modèle conceptuel est d'expliquer la signification des termes et des concepts utilisés par les experts du domaine afin de leur permettre de discuter d'un problème et de trouver la relation appropriée entre les différents concepts. En d'autres mots, le modèle conceptuel représente des entités propres à un domaine et les relations entre elles. Le modèle conceptuel tente de clarifier la signification de termes pour éliminer l'ambigüité qu'ils peuvent véhiculer. Il assure qu'il n'y aura pas de confusion entre l'interprétation des termes et des concepts. De mauvaises interprétations peuvent facilement causer l'échec d'un projet. Une fois que les concepts du domaine ont été créés et définis sans ambiguïté, ils deviennent des fondations solides pour les développements subséquents. Les concepts développés dans le cadre d'un modèle conceptuel peuvent être utilisés comme base d'un design orienté objet [38] et implantés sous forme de programme par exemple. Cette section présente certains modèles conceptuels pertinents présentés dans la littérature desquels s'inspire le modèle conceptuel présenté dans cette thèse.

Actor-Property-Interaction Architecture (APIA)

APIA [39] est un paradigme qui se différentie de l'approche classique de modélisation orientée objet [40]. Cette approche de représentation d'un monde réel en monde virtuel est centrée principalement sur les interactions entre les entités d'où le terme « *interaction-centric modeling* » qui est utilisé pour la caractériser. Cette approche est plus spécifique que la modélisation orientée objet. En effet, le concept de classe, tel que défini en modélisation orientée objet, est divisé en trois éléments : « *Property* », « *Interaction* » et « *Character* ». Ces éléments sont regroupés à l'intérieur d'un contenant nommé acteur. Ces acteurs sont reliés entre eux par des « *part-whole relationships* [41] » empruntées de la science cognitive.

Premièrement, les éléments requis pour un monde virtuel modélisé en utilisant l'approche APIA sont définis comme suit :

« *Actor* » (Acteur) est l'entité de la simulation. Il est la « chose » qui est simulée. Des exemples d'acteurs sont : un avion, un sous-marin, une souris, un pilote virtuel… L'acteur est l'équivalent d'une classe en modélisation orientée objet sauf qu'il ne possède pas de méthodes ou d'attributs.

« *Property* » (Propriété) représente les données physiques ou abstraites : masse, vélocité, position, orientation… Elles sont utilisées à la place des attributs et ne sont pas encapsulées à l'intérieur de l'acteur. Elles peuvent être utilisées à plusieurs endroits au même moment, elles peuvent posséder des comportements prédéfinis et elles ne peuvent être modifiées que par le biais d'interactions spécifiques.

« *Interaction* » (Interaction) est le lien entre les propriétés appartenant aux acteurs. Les comportements sont implantés à l'intérieur des interactions : application de la gravité entre des corps, transfert de chaleur entre deux corps, explosion d'un missile… Elles sont en quelque sorte des algorithmes qui modifient les propriétés lorsqu'appelées par le gestionnaire des événements. Les interactions connaissent les propriétés requises à l'implantation d'un comportement donné.

« *Character* » (Personnage) est un groupe de propriétés définies sous un nom spécifique. Une propriété n'est pas un personnage. Par exemple, un personnage pouvant entrer en collision (« *Collisionable* ») contient les propriétés comme sa forme géométrique et sa position dans l'espace.

Ensuite, les interconnexions entre les personnages se matérialisent par le biais d'interactions. Une interaction peut dépendre d'un ou plusieurs personnages comme l'illustre la Figure 8.

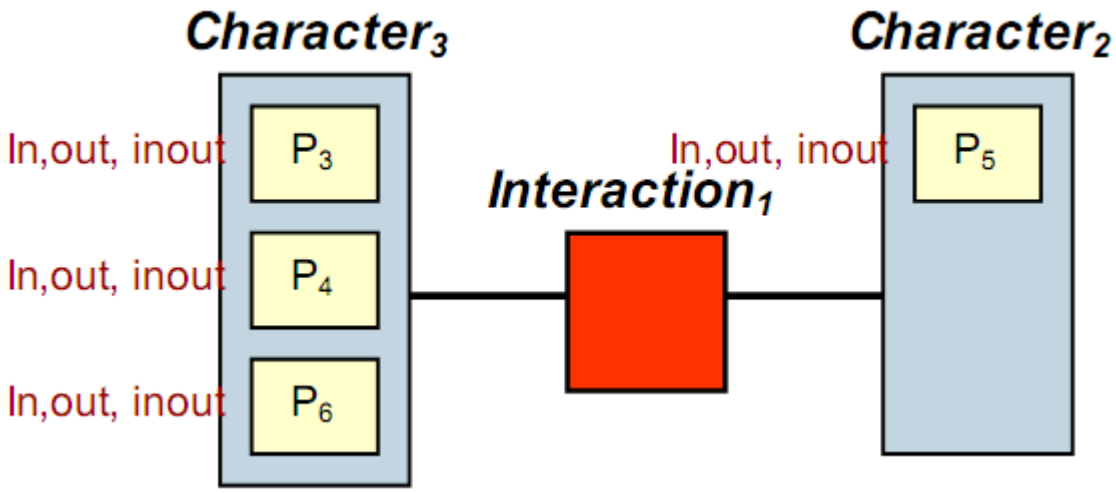


Figure 8 Association personnage-interaction

Les propriétés attachées aux personnages servent d'entrée et/ou de sortie à l'interaction.

Associer un personnage à un acteur permet à ce dernier de prendre vie à l'intérieur du monde virtuel. Comme le montre la Figure 9, l'héritage est un bon moyen pour associer un ou plusieurs personnage(s) à un acteur.

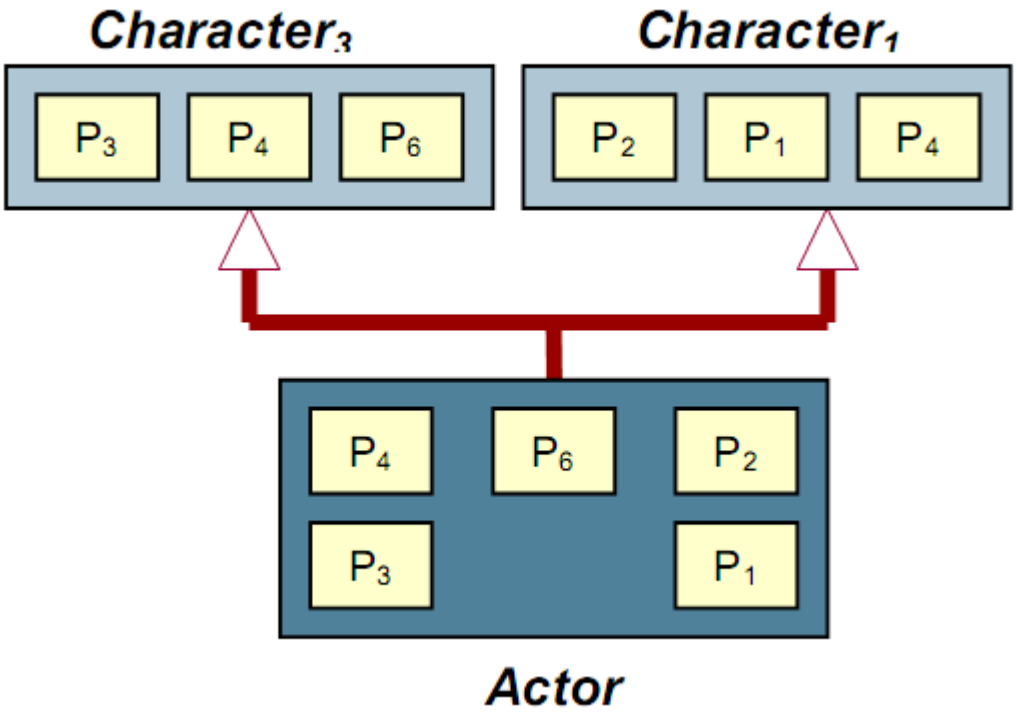


Figure 9 Assignation de caractères à un acteur

Ce type d'héritage, qui peut être comparé à celui d'inclusion de classes [41], signifie qu'un acteur est un type de caractère. En ce sens, un acteur sait comment interagir avec les autres acteurs qui héritent de personnages complémentaires. Un acteur peut agir et réagir avec d'autres acteurs parce qu'il est associé à des interactions.

L'architecture d'APIA renferme des éléments desquels peuvent s'inspirer les travaux de cette thèse en ce qui a trait au modèle conceptuel qui sera réalisé. En effet, l'atout principal d'APIA est l'introduction du concept de modélisation centrée sur les interactions. Ceci signifie que l'architecture se concentre sur le paradigme de connexion inter-entités plutôt que sur un paradigme centré sur les entités elles-mêmes. Par le fait même, l'attention s'éloigne des entités pour se concentrer plus spécifiquement sur les interactions qu'elles ont avec leur environnement ou les autres entités. De cette façon, le monde virtuel devient un ensemble plus cohérent et plus unifié car l'évolution de l'environnement n'est pas seulement basée sur l'entité elle même, mais plutôt sur les interactions qu'elle a subit avec l'environnement ou ses semblables. Ceci est un atout lors de l'étude d'un système complexe. En effet, dans le cas de l'analyse d'un système complexe, l'attention doit porter sur

l'ensemble des entités afin de permettre la détection de phénomènes d'émergence et d'auto-organisation. Par contre, la flexibilité qu'offre cette architecture, qui s'avère être une force dans le contexte sous lequel APIA a été développé, permet difficilement de détecter ou filtrer les données de simulation. En effet, un cadre plus rigide avec des entités préétablies serait plus utile pour un filtrage des données afin de comprendre plus facilement les comportements d'un système complexe simulé.

« *Pattern-Oriented Modeling (POM)* »

Cette section présente le POM en se basant sur les travaux de Wiegand et al. [42].

Le POM fut formulé tout d'abord en écologie qui est une science avec une longue tradition de modélisation du « simple au complexe » (« *bottom-up* »).

« *A bottom-up approach is the piecing together of systems to give rise to grander systems, thus making the original systems sub-systems of the emergent system* [43]. »

Le but principal de la stratégie POM est de rendre la modélisation « *bottom-up* » plus rigoureuse et compréhensive [44]. Par le fait même, POM suit l'approche classique de la recherche en science, c'est-à-dire l'explication de patrons observés. Un patron est défini comme suit:

« *Patterns are observations of any kind showing nonrandom structure and therefore containing information on the mechanisms from which they emerge.* »

Un patron permet de définir les caractéristiques d'un système. Un patron peut représenter un processus interne essentiel ou une structure interne cachée dans la simulation. Du coup, il renferme l'information sur l'organisation interne du système sous forme de code. Le but principal de l'approche POM et de décoder cette information à l'aide de patrons observés dans la nature [45].

La motivation de POM est que, pour un système complexe, un seul patron n'est pas suffisant pour réduire l'incertitude sur la structure d'un modèle et de ses paramètres. En effet, ce phénomène a été depuis longtemps observé en science. Par exemple, la règle de Chargaff concernant les paires de bases de l'ADN ne fut pas suffisante pour en décoder la structure. En effet, la combinaison de la diffraction des rayons X et les propriétés tautomèriques de la purine et de la pyrimidine ont été nécessaires pour y arriver [46]. Le but de l'approche POM est d'optimiser la complexité du modèle par l'utilisation de patrons afin de trouver la « *medawar zone* » [47]. La « *medaware zone* » est la section d'un problème où il est le plus probable de trouver une solution fructueuse. Un problème modélisé de façon trop simple n'a pas beaucoup de chances de

fournir des résultats valables. À l'opposé, une modélisation trop ambitieuse a beaucoup de chances de ne jamais voir le jour. La Figure 10 illustre le concept de la « *medawar zone* ».

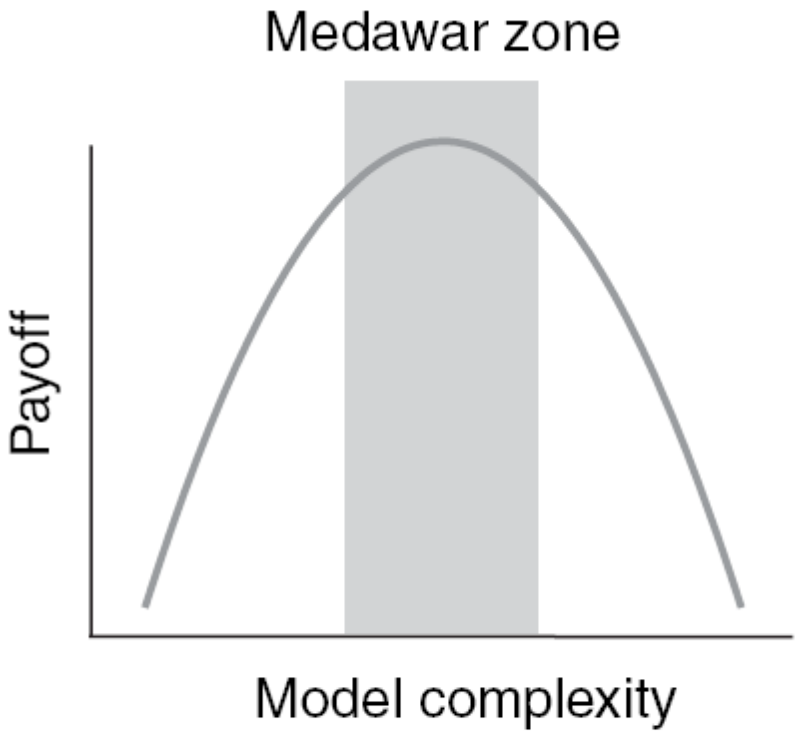


Figure 10 « *Medawar zone* » [45]

L'idée de se servir de patrons pour comprendre la simulation d'un système complexe est intéressante, car elle permet de voir les entités de la simulation sous forme de groupes interagissant entre eux. En effet, elle permet de transposer l'attention des entités vers le groupe. Par contre, elle n'est pas une approche générale qui peut résoudre plusieurs analyses de systèmes complexes. En effet, un processus de collecte de données et de validation est nécessaire pour chaque système complexe à l'étude. C'est de cette façon que les patrons sont détectés et par la suite codés. Donc, trouver de vrais patrons dans la nature pour créer des modèles est inspirant, mais des patrons généraux devraient être élaborés.

« *Agent-Based Model (ABM)* »

Un ABM (aussi connu sous le nom de « *multi-agent system* » ou « *individual-based model* ») est une classe de modèles computationnels qui a pour but de simuler les actions et les interactions entre des agents autonomes avec objectif d'analyser leurs effets sur le système global [48]. Un agent, pour sa part, est défini comme suit :

> « *An agent is just something that perceives and acts* [49] »

Le but de cette méthode est de créer une simulation suffisamment complexe qui permettra de montrer la dynamique d'un système sous observation tout en restant simple afin d'avoir une représentation traitable sur un ordinateur. D'une part, la « dynamique » est définie comme l'étude des changements et de l'évolution d'un système [50]. D'autre part, un problème est dit traitable lorsqu'il est possible d'en trouver une solution quand un nombre suffisant de ressources et une période de temps sont disponibles.

Getchell défini les caractéristiques d'un agent faisant partie d'un ABM comme suit :

- Activité : Chaque agent agit indépendamment selon les règles de la simulation et son comportement préprogrammé. Ces règles et comportements peuvent avoir une ou plusieurs de ces caractéristiques :
    - Orienté vers un but : Un agent peut agir d'une façon à atteindre un but précis. Par exemple, un agent peut avoir comme objectif de maximiser l'accumulation d'une certaine ressource;
    - Réaction/perception : L'agent apprend son environnement ou peut être en possession d'une carte de l'environnement. Par exemple, un agent peut être informé des endroits où se trouvent certaines ressources;
    - Rationalité limitée [51] : Généralement, les agents orientés vers un but opèrent selon le principe de choix rationnel qui implique l'accès illimité à l'information et aux ressources de calcul. Par contre, les expérimentations montrent que les décisions non-optimales sont le plus souvent celles qui ressemblent à la réalité. Donc, dans le but d'être capable de prédire le comportement des agents, ceux-ci sont contraints en ce qui à trait à l'information qui leur est accessible et à leur capacité à analyser cette information. Par exemple, un agent pourrait seulement être en mesure d'observer les ressources qui se trouvent dans un certain rayon d'action;

- Interaction : Toujours avec le principe de réalité limitée, les agents peuvent interagir ou échanger de l'information entre eux. Ces interactions peuvent avoir de l'effet sur l'agent incluant sa destruction ou un changement dans son but;
- Mobilité : L'interaction avec l'environnement et les autres agents est grandement améliorée si l'agent peut parcourir l'espace de manière autonome;
- Adaptation : L'altération de l'état de l'agent par ses interactions avec l'environnement ou un autre agent fournit une forme d'apprentissage ou de mémoire. Cette adaptation peut s'effectuer au niveau de l'agent, d'un groupe d'agents proches ou de la population entière des agents.

- Autonomie : Chaque agent est libre de prendre ses propres décisions.
- Hétérogénéité : Même si un agent peut être membre d'un groupe ayant des éléments en commun, il se développe de façon autonome.

La Figure 11 montre la représentation générique d'un agent. Ce dernier agit (« *actions* ») sur l'environnement en fonction des perceptions (« *percepts* ») qu'il a de celui-ci via l'utilisation de capteurs (« *sensors* »).

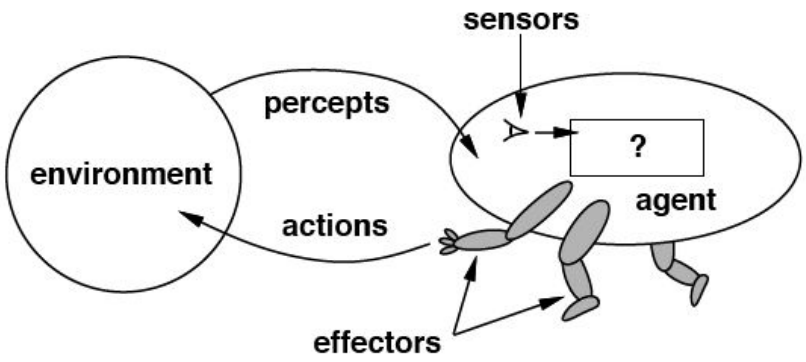


Figure 11 Représentation générique d'un agent [49]

L'approche ABM propose des idées qui ont inspiré les travaux de cette thèse. En effet, lorsqu'il est question d'individus évoluant dans un système complexe, cette approche résume bien de quelle façon il doit être modélisé. Premièrement, l'agent est autonome. Alors, il possède toutes les propriétés dont il a besoin pour évoluer dans l'environnement. Deuxièmement, chaque agent agit en fonction de règles préétablies et indépendamment des autres. Par contre, son action peut altérer l'état d'autres agents ou changer l'environnement. Finalement, même si les agents peuvent avoir une base préprogrammée commune, ils

évoluent tous de façon autonome ce qui crée un environnement hétérogène. ABM traite donc les agents indépendamment ce qui favorise leur évolution. Par contre, pour analyser les effets de groupes, d'autres approches basées sur l'ensemble des agents doivent être utilisées, car ABM ne se concentre que sur l'individu et son évolution et non sur l'évolution de l'ensemble des individus

Résumé

La revue de la littérature sur les modèles conceptuels a permis de faire une synthèse des contributions qui pourraient inspirer les travaux réalisés dans cette thèse. Premièrement, l'approche ABM a permis de voir que lorsqu'un système complexe est simulé, il est préférable de centrer l'évolution du système sur les individus séparément. De cette façon, ils peuvent prendre leurs propres décisions et évoluer dans l'environnement de façon autonome. Deuxièmement, APIA montre que de centrer la modélisation sur les interactions permet de créer une meilleure cohérence entre les individus d'un environnement virtuel. En effet, pour comprendre un problème dans son ensemble il faut fixer l'attention sur les entités elles-mêmes et regarder le problème dans son ensemble. C'est ce que permet la modélisation basée sur les interactions dans APIA. Finalement, le POM révèle que lors de simulations de systèmes complexes, des patrons de comportements ou des regroupements d'individus peuvent émerger. Ces patrons permettent de mieux comprendre ce qui se produit en analysant les regroupements d'individus. Ceci cadre avec les systèmes complexes qui sont caractérisés par l'auto-organisation et l'émergence de comportements.

La détection d'interactions entre entités de la simulation

Cette section de l'état de l'art se concentre sur le deuxième sujet du projet qui nous intéresse dans cette thèse, c'est-à-dire la détection d'interactions entre les entités de la simulation. La compréhension d'un système complexe passe par l'analyse de l'information fournie par ce dernier. Ce sont les données représentant le système simulé qui permettent de détecter l'auto-organisation ou l'émergence de comportements. Pour permettre de détecter ces aspects des systèmes complexes, des outils doivent être développés afin de faciliter l'intégration de l'information par l'utilisateur. Malheureusement, ce ne sont pas les données brutes qui permettent de bien comprendre le système, ce qui est particulièrement vrai en simulation temps réel. En effet, ce sont les statistiques descriptives des données brutes mises ensemble par des algorithmes de détection d'interactions qui permettent d'informer le plus adéquatement les utilisateurs. La détection d'interactions entre les entités de la simulation est donc un outil intéressant pour mieux comprendre les systèmes complexes. Les techniques pour détecter les interactions seront présentées dans les prochaines sections.

La logique floue

> « Humans have a remarkable capability to reason and make decisions in an environment of uncertainty, imprecision, incompleteness of information, and partiality of knowledge, truth and class membership. The principal objective of fuzzy logic is normalization/mechanization of this capability.[52] »

C'est dans le but de reproduire la capacité humaine de raisonner dans un environnement d'incertitude que la logique floue [53] est utilisée. En effet, selon ce concept, un élément peut appartenir plus ou moins fortement à une classe ce qui est appelé le « *degree of truth* » [54]. La Figure 12 montre un exemple de système de décision par logique floue. Les différentes étapes de la logique floue y sont présentées. Le système permet de décider combien un client doit laisser de pourboire au restaurant en fonction de la qualité du service et de la qualité de la nourriture.

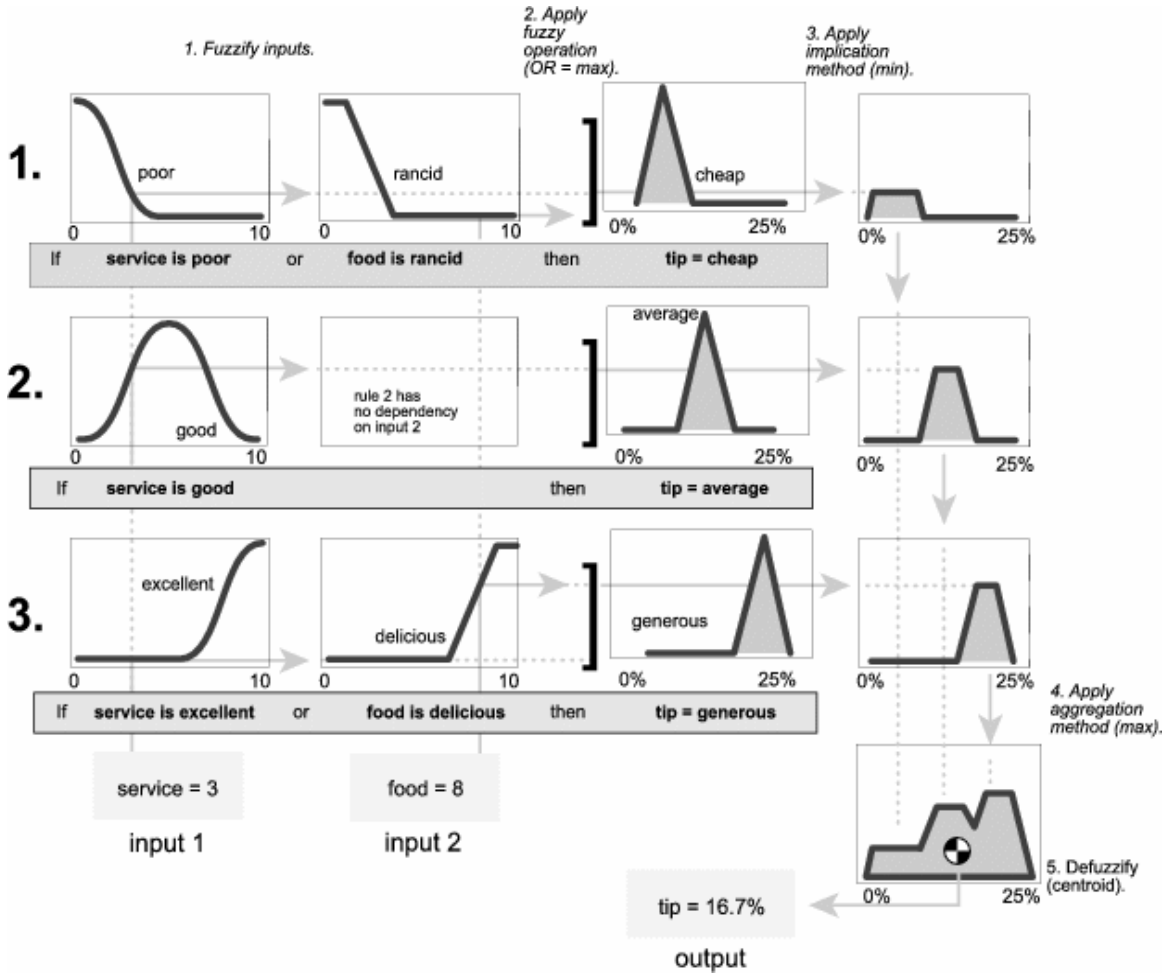


Figure 12 Exemple de système de décision à base de logique floue [55]

La première étape (« *fuzzify inputs* » sur la Figure 12) consiste à prendre les entrées et à déterminer leur degré d'appartenance à chaque ensemble flou par le biais de fonctions membres. Après cette première étape, le degré d'appartenance à chaque règle est connu. Cette partie se nomme l'antécédent. Dans le cas ou l'antécédent est composé de plus d'une partie (comme sur la ligne 1 et 3 de la Figure 12), un opérateur logique (« *Apply fuzzy operation* » sur la Figure 12) est appliqué pour obtenir un seul nombre pour la règle en question qui représente le conséquent. Ensuite, la troisième étape consiste à appliquer la méthode d'implication (« *Apply implication method* » sur la Figure 12). La fonction d'implication utilisée dans l'exemple est le *minimum*. Par la suite, la quatrième étape combine (« *Apply aggregation method* » sur la Figure 12) les résultats de l'étape 3 afin de pouvoir prendre une décision finale. Finalement, la dernière étape est la defuzzification (« *Defuzzify* » sur la Figure 12). Elle consiste, dans l'exemple, à prendre le centroïde du résultat de l'agrégation comme résultat final. Lorsque le service est de 3/10 et la nourriture de 8/10, un pourboire de 16.7% est adéquat.

La logique floue permet le raisonnement approximatif, ce qui peut être grandement utile lorsqu'il est temps de détecter une interaction entre entités, car, pour la plupart des systèmes réels rien n'adopte généralement un caractère binaire.

La logique floue probabiliste (PFL)

L'article « *Interaction Detection via Probabilistic Fuzzy Logic for Coupled Dynamical Systems* [56] » propose la détection des interactions entre des systèmes dynamiques couplés avec l'aide de la logique floue probabiliste (« *probabilistic fuzzy logic* »)[57]. Les grandes lignes de cette approche sont discutées ci-dessous.

La grande différence entre les PFR et les règles floues standards présentées à la section « La logique floue » est que chaque possibilité d'entrée peut correspondre à chaque valeur de sortie floue. Donc, au lieu d'avoir une règle standard de type « *if-then »* :

IF *variable* IS *property* THEN *action*

où *variable* est l'entrée, « *property »* est la valeur de l'entrée et *action* est la sortie, la PFR à une série de règles « *if-then »* du type :

IF *variable* IS *property* THEN *action with a probability(1)*

IF *variable* IS *property* THEN *action with a probability(2)*

…

IF *variable* IS *property* THEN *action with a probability(N)*

où chaque action est possible pour chaque entrée avec une probabilité *« probability(x) »* donnée. La sortie du système de logique floue n'est plus un index vers une sortie particulière mais plutôt un vecteur de possibilités de sorties avec des probabilités associées. Pour obtenir une valeur de sortie unique du système une sélection aléatoire est effectuée.

L'algorithme proposé pour concevoir un système basé sur la logique floue probabiliste tel qu'utilisée dans l'article afin de détecter des interactions referme trois étapes.

1. La première étape est la même que pour la logique floue standard c'est-à-dire la « *fuzzyfication* ». Pour chaque entrée et sortie nous définissons au moins deux fonctions membres. Par la suite, une assignation qualitative leur est associée comme, par exemple, petit, moyen et grand (Figure 13). La forme de ces fonctions membres peut être diverse mais, la plupart du temps, elles sont représentées par des triangles et des trapézoïdes.

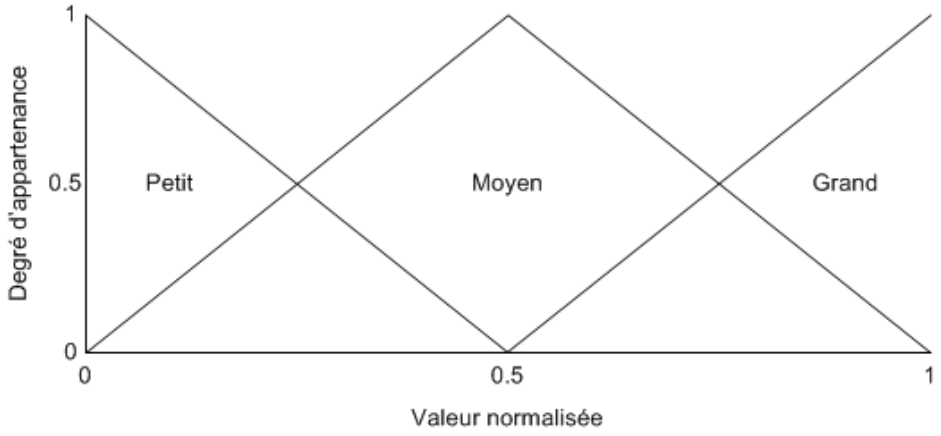


Figure 13 Degré d'appartenance en fonction de la valeur normalisée pour les classes petit, moyen et grand

2. La seconde étape consiste à calculer les probabilités pour les combinaisons d'entrées et de sortie et pour les entrées individuellement. D'une part, chaque combinaison possible d'entrées est associée à chacune des sorties au lieu d'être associée à une seule sortie comme pour la logique floue standard. Il en résulte que la valeur de sortie pour chaque combinaison d'entrées est un vecteur de probabilités. La valeur des probabilités peut être calculée avec une équation proposée dans l'article.

D’autre part, les probabilités d’obtenir une sortie pour une seule entrée sont calculées. Ces valeurs peuvent être obtenues par l’analyse de la série temporelle de données de l’entrée en question. Donc, à la fin de cette étape, les probabilités pour la combinaison de deux entrées sont connues ainsi que les probabilités pour une seule entrée.

3. La dernière étape consiste à calculer la distance entre la probabilité combinée des deux entrées et celle de la probabilité d’une seule de deux entrées. La distance utilisée dans l’article est celle de Kullback-Leibler [58]. De cette façon il est possible de calculer le degré de chance que la première série temporelle soit influencée par la seconde.

Cet article montre qu’il est possible de détecter les interactions entre les entités d’un système faiblement couplé par l’utilisation de la logique floue probabiliste. Ce cas particulier de la logique floue implique par contre que chaque possibilité de sortie peut survenir avec une entrée donnée. De plus, des séries temporelles pour le système en question doivent être connues afin de pourvoir calculer les probabilités.

Réseaux de neurones artificiels

Un réseau de neurones artificiels (Figure 14), communément appelé réseau de neurones, est un modèle mathématique ou computationnel qui s’inspire de la structure et/ou des aspects fonctionnels des réseaux de neurones biologiques [59]. Il consiste à une interconnexion de neurones artificiels (Figure 15) et traite l’information grâce aux connexions. Il est à noter que le connexionnisme modélise les phénomènes mentaux ou les comportementaux comme des processus émergents de réseaux d'unités simples interconnectées [60].

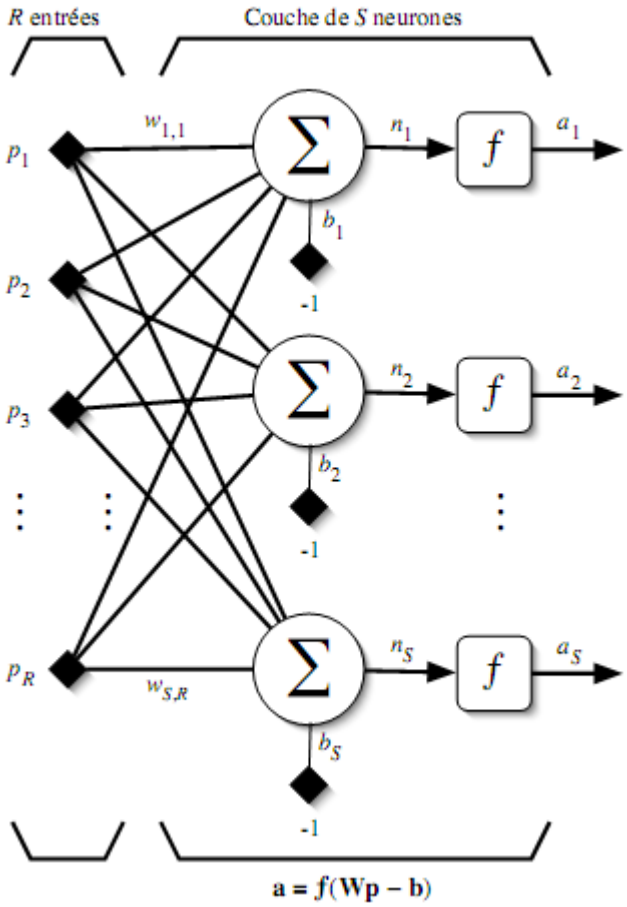


Figure 14 Réseau de neurones [61]

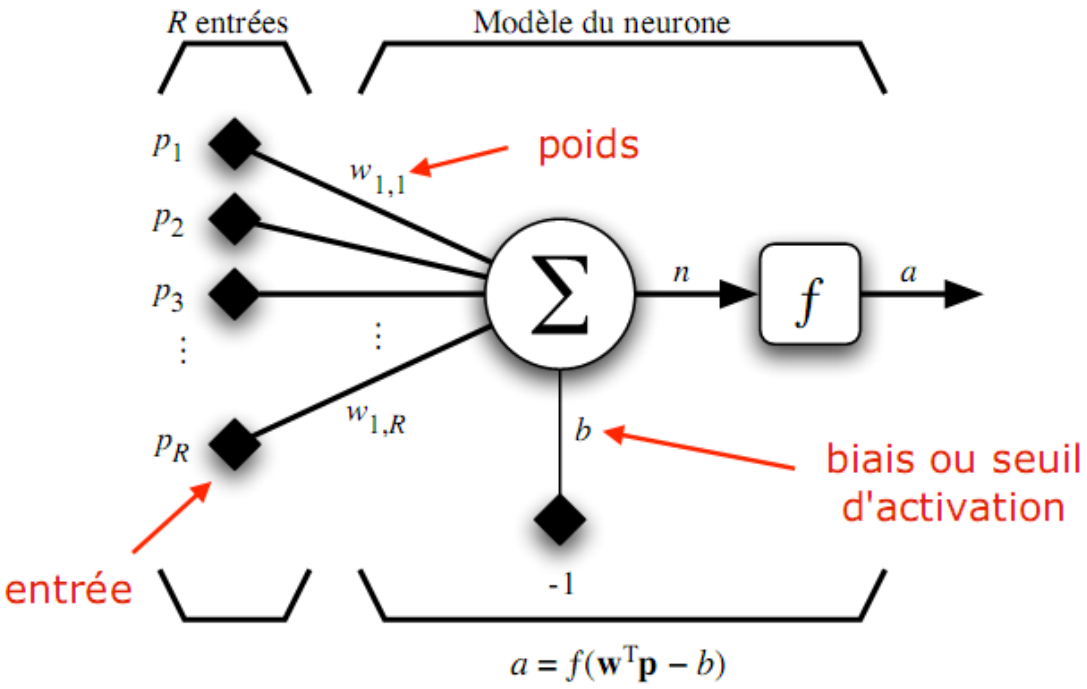


Figure 15 Neurone artificiel [61]

Dans la plupart des cas, le réseau de neurones est un système adaptatif qui change sa structure en fonction de l'information externe ou interne transitant à travers le réseau lors d'une phase d'apprentissage. En effet, une série de paires entrées/sorties est utilisée lors de la phase d'apprentissage. Ces paires sont obtenues au préalable à partir d'observations. L'algorithme ci-dessous permet à un réseau de neurones de type *perceptron* [61] d'apprendre à partir d'observations :

1. Initialiser tous les poids à de petites valeurs aléatoires dans l'intervalle [−0.5, 0.5];
2. Normaliser les données d'entraînement;
3. Permuter aléatoirement les données d'entraînement;
4. Pour chaque donnée d'entraînement n :
    a. Calculer les sorties observées en propageant les entrées vers l'avant ;

    Ajuster les poids en rétropropageant l'erreur observée :

$$wji(n) = wji(n-1) + \Delta wji(n) = wji(n-1) + \eta\delta j\,(n)yi(n)$$

où le « gradient local » est défini par :

$$\delta j\,(n) = \begin{cases} e_j(n)y_j(n)\left[1 - y_j(n)\right] & Si\ j \in couche\ de\ sortie \\ y_j\,(n)[1 - y_j\,(n)]\sum_k \delta_k(n)w_{kj}(n) & Si\ j \in couche\ cachée \end{cases}$$

avec $0 \leq \eta \leq 1$ représentant le taux d'apprentissage et $y_i(n)$ représentant soit la sortie du neurone i sur la couche précédente, si celui-ci existe, soit l'entrée i autrement.

5. Répéter les étapes 3 et 4 jusqu'à ce qu'un nombre maximum d'itérations soit atteint ou jusqu'à ce que la racine de l'erreur quadratique moyenne (EQM) soit inférieure à un certain seuil.

Les réseaux de neurones modernes sont des outils pour modéliser des données statistiques non-linéaires. En effet, ils sont utilisés pour modéliser des relations complexes entre les entrées et sorties d'un système à l'étude. D'une part, ils peuvent servir de mémoire par l'assimilation d'une grande quantité d'exemples préalablement traités lors de la phase d'apprentissage. Par le fait même, lorsque le réseau sera soumis à une entrée déjà traitée, il pourra assigner correctement la sortie. D'autre part, ils sont aptes à traiter des entrées ne faisant pas partie des exemples utilisés lors de la phase d'apprentissage. De cette façon, une nouvelle entrée similaire à celles traitées en apprentissage donnera une sortie valable [62].
L'inconvénient majeur de réseaux de neurones est qu'une quantité suffisante et représentative de données doit être recueillie afin de permettre l'entraînement du réseau. Dans les cas où le domaine de recherche est en exploration il est parfois difficile de recueillir suffisamment de données pour que la phase d'entraînement s'exécute correctement.

Arbres de décision

Un arbre de décision (Figure 16) est un outil aidant à la prise de décisions. Il utilise un arbre pour modéliser les décisions avec leurs conséquences [63]. Il est composé d'un ensemble de règles simples de type : « si le sépale est de 5.45cm ou moins, le spécimen est un setosa ». Donc, chaque nœud de l'arbre correspond à un

test pour une valeur d'une des propriétés. Pour leur part, les branches représentent les valeurs possibles résultant du test [49].

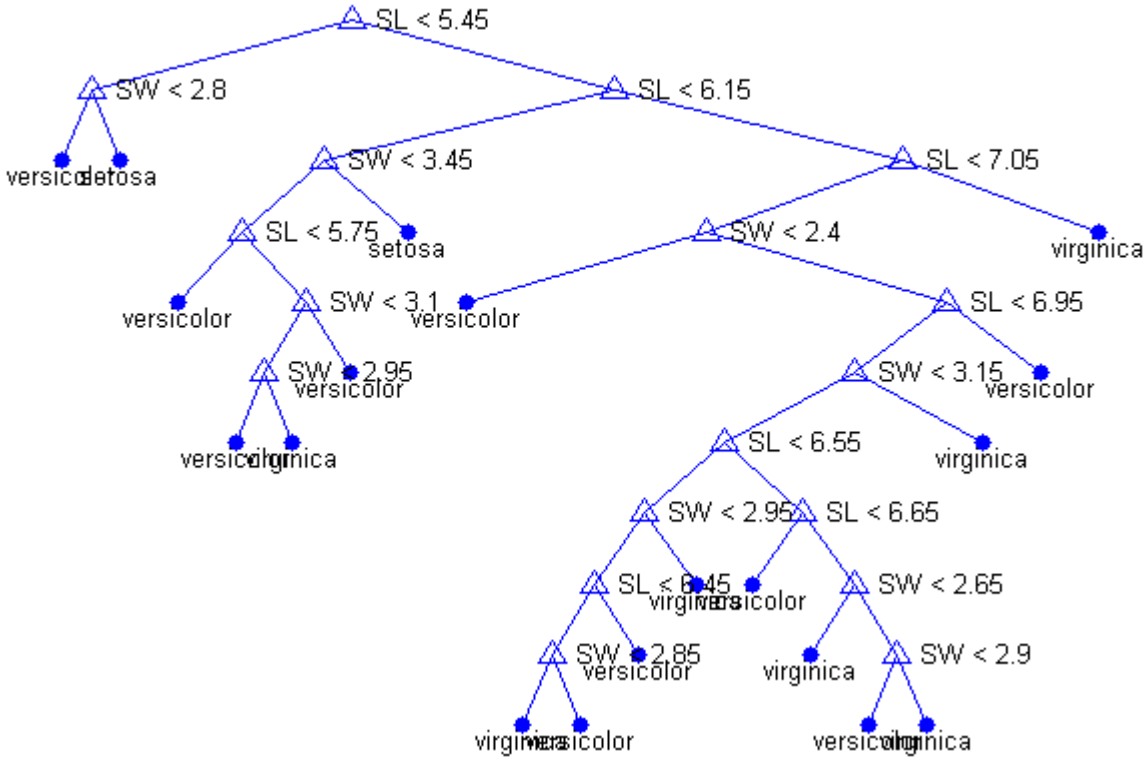


Figure 16 Arbre de décision pour classifier les sortes d'iris [64]

Cet outil est approprié pour les problèmes de classification. L'exemple suivant démontre de quelle façon il permet de classifier les iris en fonction des dimensions de leur sépale. Un ensemble de 150 iris a été étudié afin de créer le classificateur. Les iris ont été répertoriés en fonction de la largeur et de la longueur de leur sépale (Figure 17). Trois ensembles sont représentés : setose, versicolor et virginica.

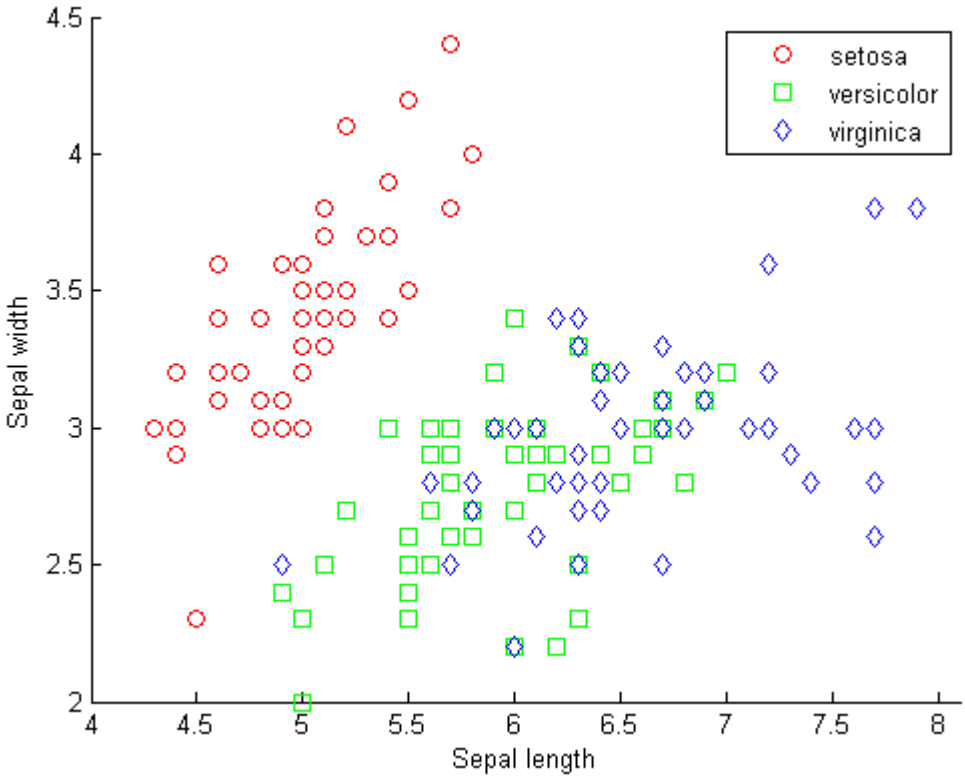


Figure 17 Ensembles connus d'iris avec la dimension de leur sépale[64]

Par l'utilisation d'une fonction de Matlab *classregtree* il est possible de diviser le plan afin de représenter les trois classes (Figure 18) et de créer l'arbre de décision (Figure 16).

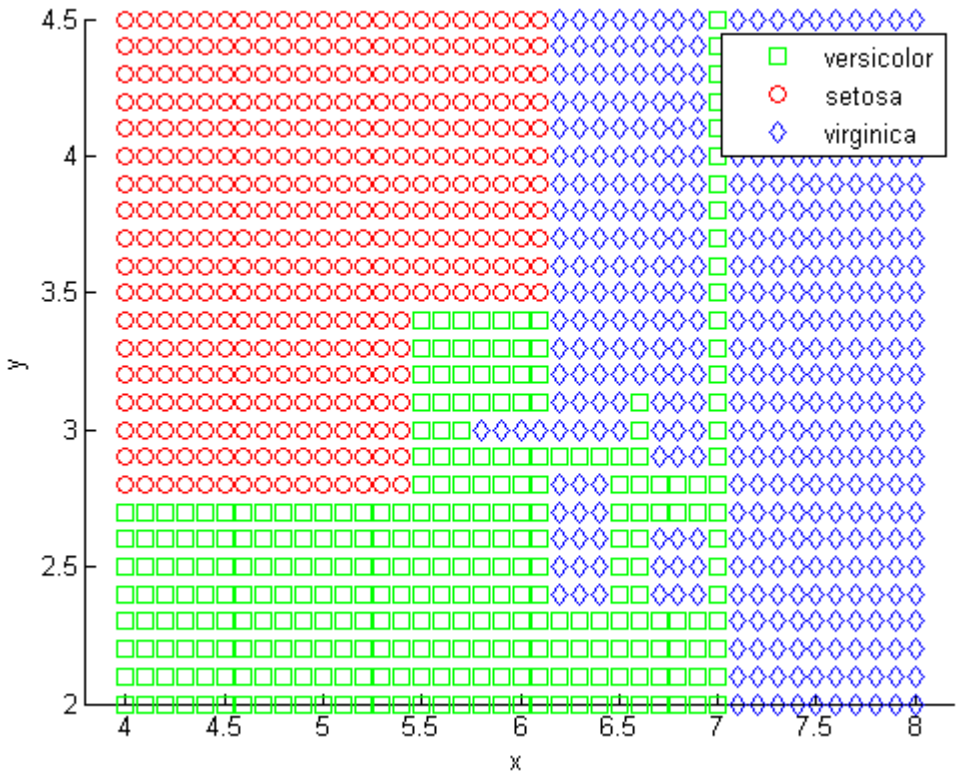


Figure 18 Plan divisé en classes pour les types d'iris[64]

De cette façon, au moment de catégoriser un nouvel échantillon d'iris, il suffit de suivre l'arbre de décision afin de lui assigner une classe en fonction de la longueur (LS) et de la largeur (WS) de son sépale.

Cette méthode renferme de nombreux avantages [63] :

1. est simple à comprendre et à interpréter;
2. elle est valable même avec peu de données brutes. En effet, l'utilisation d'un expert du domaine à l'étude peut permettre de créer un arbre dans sa quasi-totalité;
3. elle utilise le modèle de la *white box*. Il est donc facile de comprendre le pourquoi d'un résultat en regardant l'arbre qui a été utilisé.

Il est à noter que l'arbre de décision est non-paramétrique ce qui signifie qu'il n'est pas nécessaire de connaître la distribution statistique de chaque classe pour le construire.

Résumé

Les techniques de détection d'interactions ont toutes leurs forces et leurs faiblesses. Le réseau de neurones est très efficace pour trouver la relation non-linéaire entre les entrées et les sorties d'un système. Par contre, il requiert une phase d'entraînement qui exige une grande quantité de données brutes. Dans les cas où le système n'est pas disponible pour faire l'acquisition des données, cette méthode n'est pas appropriée.

L'arbre de décision permet de palier au manque de données brutes par l'utilisation d'un expert. Par contre, à chaque nœud, la décision est prise de façon franche. Le flou n'est pas inclus dans cette méthode. Pour sa part, la logique floue incorpore un caractère incertain à la prise de décisions. De plus, la combinaison de données brutes et l'utilisation d'un expert sont suffisantes pour créer un système basé sur la logique floue. Cette technique referme les caractéristiques principales recherchées dans le cadre de notre recherche pour la détection d'interactions.

Chapitre 2 : Modèle conceptuel et architecture

Ce chapitre sera consacré à la description du modèle conceptuel qui sera utilisé et à l'architecture du Framework qui en découle. D'une part, un modèle conceptuel sera établi afin de clarifier et définir les termes qui seront utilisés lors du développement du Framework et d'établir un vocabulaire du domaine. De plus, le modèle conceptuel permettra de définir les entités et les relations entre entités dans le but de faciliter la création de l'architecture logicielle. D'autre part, l'architecture qui représente un ensemble de structures nécessaire à la réalisation du Framework sera décrite.

Modèle conceptuel

Cette section décrit le modèle conceptuel qui a été développé. Par définition, un modèle conceptuel vise à définir les entités et les relations entre elles. Ceci permet aux utilisateurs et aux experts de partager le même vocabulaire relatif aux problèmes d'assistance de simulation visuelle interactive de systèmes complexes par la détection d'interactions. D'une part, l'origine du modèle conceptuel sera expliquée. D'autre part, la définition des termes du modèle conceptuel développé sera présenté.

Origine

Pour permettre l'aide à l'analyse en cours de simulation de systèmes complexes, un modèle conceptuel est proposé. Il permet de standardiser le vocabulaire de la nouvelle approche présentée dans cette thèse qui consiste à la détection d'interactions entre les entités simulées en se basant sur le paradigme des écosystèmes. Cette section présente les implications de ce modèle conceptuel et décrit son origine. D'une part, les raisons justifiant le développement du modèle conceptuel sont présentées. D'autre part, les inspirations principales qui ont mené à la création du modèle conceptuel sont survolées.

Lors de la réalisation de l'état de l'art sur la simulation des systèmes complexes, une lacune a été notée en ce qui à-trait à l'analyse en cours de simulation. Dans la majorité des cas, l'utilisateur est laissé à lui-même lorsque vient le temps d'intégrer l'information générée par le modèle simulé. Ce qui implique que l'utilisateur peut être surchargé au plan cognitif. En effet, un système complexe renferme de nombreux paramètres qui interagissent de façon non-linéaire. Par le fait même, la nouvelle information générée par l'interaction des entités doit être assimilée rapidement par l'utilisateur afin de lui permettre de bien suivre la simulation. Une question doit donc être abordée : est-ce que l'ajout d'un mécanisme d'analyse entre le modèle simulé et

l'utilisateur peut être bénéfique à la compréhension? Ce problème a été évalué par l'ajout d'un système de détection, en cours de simulation, d'interactions se basant sur le paradigme des écosystèmes.

La principale raison justifiant l'approche proposée est de permettre la réalisation d'expériences de simulation visuelle interactive de systèmes complexes qui ne surchargeront pas l'utilisateur au plan cognitif par la grande quantité d'information disponible dans de tels contextes. Afin que l'approche proposée atteigne ses objectifs, il est utile d'élaborer un modèle conceptuel qui définira un vocabulaire et un ensemble de relations entre les concepts couverts par le vocabulaire. Le modèle conceptuel peut s'inspirer de nombreux domaines pour établir le vocabulaire et orienter l'approche de solution à l'exploration de situations complexes. La nature même d'un système complexe, comme présenté à la section « Systèmes complexes », met l'emphase sur le fait que la compréhension passe par les interactions entre les parties qui le composent. Ce qui implique que de s'appuyer sur les bases d'un domaine fortement établi depuis longtemps[65] qui a comme objectif l'étude des interactions et leurs conséquences entre les organismes serait judicieux. L'écologie est un tel domaine qui répond à ces spécifications.

> « L'écologie est l'étude scientifique des interactions entre les organismes d'une part et entre les organismes et leur milieu d'autre part, dans les conditions naturelles. » [66]

Le domaine de l'écologie permet donc d'établir les fondations d'un vocabulaire pour le modèle conceptuel qui sera réalisé dans le cadre des travaux de recherches, car il possède les éléments recherchés pour représenter les concepts rencontrés dans l'étude des situations complexes.

Vocabulaire du modèle conceptuel proposé

La définition du vocabulaire d'un modèle conceptuel est essentielle, car elle permet de clarifier les termes qui lui sont associés. Par le fait même, le vocabulaire permet d'éliminer les ambiguïtés qui pourraient survenir. Cette section présentera le vocabulaire qui sera utilisé dans le cadre du projet de thèse. D'une part, les éléments empruntés au domaine de l'écologie seront définis et adaptés au contexte plus spécifique de la simulation de systèmes complexes. D'autre part, les interactions qui sont à la base même de la compréhension de l'écologie seront présentées et adaptées pour représenter les interactions possibles lors de la simulation de tels systèmes.

Un **écosystème** est un ensemble dynamique formé par les organismes potentiellement interactifs d'une communauté et par les facteurs abiotiques avec lesquels ils interagissent [66]. Les facteurs **abiotiques** sont

les éléments non-vivants dans un écosystème. Dans notre cas, ces facteurs seront décrits par le terme « ressource ».

Une **ressource** (ressource naturelle) est un bien, une substance ou un objet présent dans la nature, et exploité pour les besoins d'une société humaine [67]. Dans le cadre du projet, une ressource sera tout élément épuisable consommé par les organismes.

Un **organisme** est un système complexe évolutif formé d'organes qui interagissent de façon à ce qu'ils fonctionnent comme un ensemble stable [68]. Par contre, dans le présent travail, un organisme est vu avec un sens plus large. C'est-à-dire qu'un organisme représente tout élément de la simulation qui peut interagir avec d'autres éléments.

Une **espèce** est un groupe de populations qui ont le potentiel de s'accoupler dans la nature [66]. Une population est un groupe localisé d'organismes de la même espèce à un moment donné. Dans le cadre de la thèse, une espèce représente l'ensemble des organismes ayant les mêmes propriétés et un comportement semblable.

Une **propriété** est le propre, la qualité particulière de quelque chose [69]. Les propriétés sont les caractéristiques propres aux organismes. Il est à noter qu'elles sont à la base du Framework proposé. En effet, ce sera par leur analyse que les interactions seront détectées. Les propriétés seront au nombre de trois soit : la position, le rayon d'action et l'état de santé. La **position** représente la position cartésienne d'un organisme dans l'écosystème simulé. Le **rayon d'action** représente la zone d'influence d'un organisme à un moment donné. Il peut par exemple, représenter son champ de vision. Il est à noter que le rayon d'action peut, comme les autres propriétés, évoluer en cours de simulation. L'**état de santé** représente l'énergie restante pour un organisme. Dans la plupart des cas, l'état de santé est gradué de 0% à 100%. 100% signifie que l'organisme est dans sa condition optimale et 0% signifie qu'il cesse d'exister. Les éléments entrant dans le calcul de l'état de santé sont laissés au soin de l'utilisateur.

L'habitat est un concept utilisé dans le domaine de l'écologie pour décrire l'endroit dans lequel une population d'individus d'une espèce donnée ou d'un groupe d'espèces peut normalement vivre et s'épanouir [70]. Dans le cadre du projet, un habitat est l'espace occupé par une espèce. Cet espace tient compte du rayon d'action et de la position de chaque organisme. Le calcul de l'habitat sera couvert à la section « Calcul de l'Habitat » du « Chapitre 3 : Détection d'interactions ».

Les agissements des organismes composant une espèce peuvent la rendre nuisible, bénéfique ou neutre vis-à-vis une autre espèce. Les interactions sont donc définies en fonction de l'impact de chacune des espèces impliquées. Le Tableau 1 présente les interactions biologiques qui seront utilisées dans l'implantation du Framework. L'utilisation de ces termes permettra qualifier les interactions détectées par les algorithmes d'analyse de simulations.

Tableau 1 Interactions biologiques [6]

		Espèce X		
		néfaste	**neutre**	**bénéfique**
Espèce Y	**néfaste**	Compétition	Amensalisme	Prédation
	neutre	Amensalisme	Neutralisme	Commensalisme
	bénéfique	Parasitisme / Prédation	Commensalisme	Symbiose, Mutualisme

- La **compétition** est la rivalité entre espèces vivantes pour l'accès aux ressources du milieu.
- L'**amensalisme** est une interaction biologique interspécifique (entre deux espèces différentes) dans laquelle une espèce inhibe le développement de l'autre.
- La **prédation** se produit est lorsqu'un organisme d'une espèce met à mort des proies pour s'en nourrir ou pour alimenter sa progéniture.
- La relation de **neutralisme** entre des espèces est le fait qu'ils cohabitent sur un même territoire sans exercer d'influence entre elles.
- Le **commensalisme** se produit lorsqu'un hôte[2] d'une espèce fournit une partie de sa propre nourriture au commensal d'une autre espèce : il n'obtient en revanche aucune contrepartie évidente de ce dernier (le bénéfice de cette relation n'est pas réciproque).
- Le **mutualisme (ou symbiose)** est une interaction dans laquelle le symbiote[3] et l'hôte tirent tous les deux profits de cette relation qui est obligatoire pour les deux espèces (bénéfices réciproques). [6]

[2] Organisme, organe ou cellule qui abrite un parasite, qui constitue le milieu duquel le parasite tire sa subsistance et dans lequel il se multiplie [94].

[3] Organisme qui vit en symbiose avec un autre [94].

Les termes empruntés du domaine de l'écologie doivent maintenant être représentés par le modèle conceptuel proposé. Les sections suivantes présentent l'agencement statique (« Structure statique d'un écosystème ») et dynamique (« Structure dynamique d'un écosystème ») des composants définis dans le vocabulaire.

Structure statique d'un écosystème

La structure statique permet d'obtenir un aperçu des différents éléments présents dans l'écosystème à l'étude. En effet, elle permet de connaître l'agencement des organismes au sein des espèces ainsi que la valeur de leurs propriétés. La Figure 19 présente un exemple de structure statique pour un écosystème.

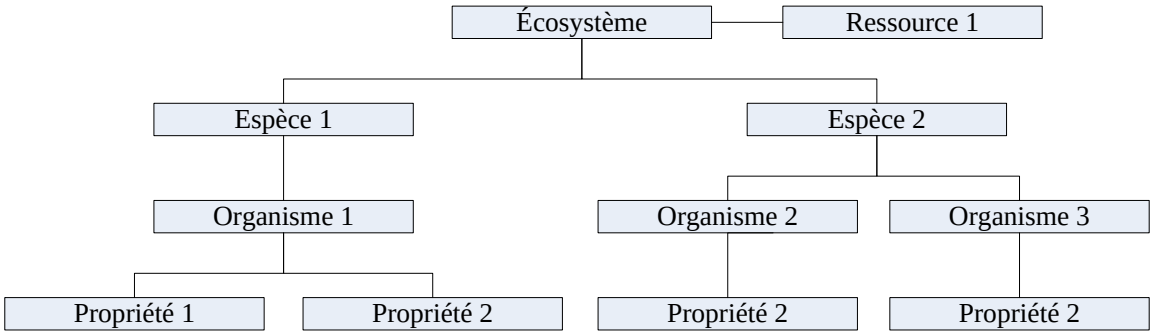


Figure 19 Exemple de structure statique d'un écosystème

La représentation de la structure statique d'un écosystème prend la forme d'un arbre [71]. En effet, chaque nœud de l'arbre représente un élément de l'écosystème. Le nœud d'un arbre renferme certaines caractéristiques qui permettent une meilleure compréhension de la représentation statique. Premièrement, un nœud peut posséder plusieurs enfants, mais ne possède qu'un seul parent. Ensuite, un nœud ne possédant pas d'enfant est un nœud de type feuille qui représente soit une propriété ou une ressource. Finalement, un parent est composé de ses enfants, ce qui signifie qu'un parent est défini par ses enfants et il n'existerait pas sans eux sauf pour les nœuds de type feuille.

La Figure 19 montre un exemple d'écosystème. Dans ce cas, un écosystème est composé d'une ressource et de deux espèces. La première, l'*espèce 1*, est composée d'un seul organisme (*organisme 1*). Pour sa part, l'*espèce 2* est composée de deux organismes soit l'*organisme 2* et l'*organisme 3*. Pour compléter l'arbre, les propriétés sont insérées comme enfant des organismes. Dans l'exemple, deux propriétés sont visibles (*propriété 1* et *propriété 2*). La *propriété 2* pourrait, par exemple, représenter la position de l'organisme dans l'écosystème. Chaque organisme aurait donc une position unique.

La structure statique est intéressante pour illustrer la hiérarchie de l'écosystème ainsi que les valeurs des différentes propriétés associées aux organismes. Par contre, une « représentation » doit être ajoutée à la structure statique afin d'établir l'aspect dynamique d'un écosystème. Une telle représentation sera expliquée dans la section suivante.

Structure dynamique d'un écosystème

Le dynamisme d'un écosystème peut être défini par les différentes interactions présentes entre les espèces. C'est pour cette raison qu'une représentation visuelle capturant l'essence même du dynamisme d'un écosystème, les interactions, a été réalisée et est présentée dans cette section.

La Figure 20 montre un exemple de structure dynamique pour un écosystème.

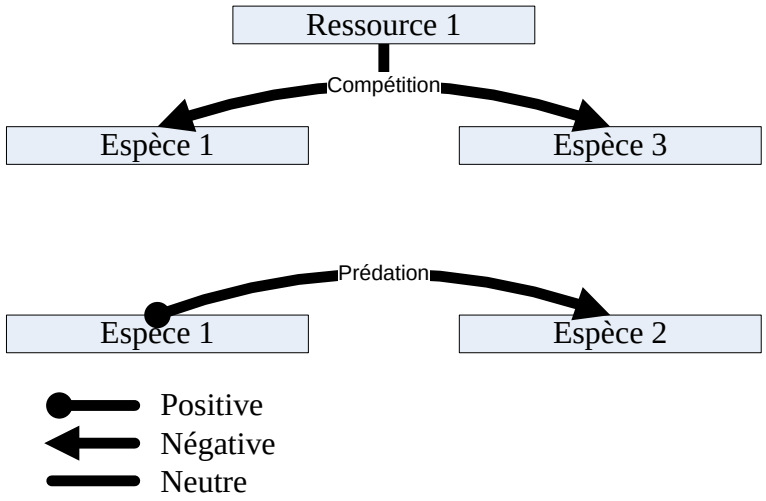


Figure 20 Exemple de structure dynamique d'un écosystème

La Figure 20 est composée de deux éléments : des boîtes et des flèches. D'une part, les boîtes représentent les éléments ayant une présence physique dans l'écosystème c'est-à-dire les espèces et les ressources (voir Figure 19 pour la hiérarchie d'un écosystème). Dans le cas présent, trois espèces existent et peuvent interagir entre elles (*espèce 1*, *espèce 2* et *espèce 3*). De plus, une ressource (*ressource 1*) est impliquée dans une interaction. En effet, une ressource peut être la source d'une interaction de compétition. D'autre part, les liens représentent les interactions (voir le Tableau 1 pour les différents types d'interactions) entre les espèces. Les terminaisons du lien indiquent le type d'influence que l'interaction à sur l'espèce. Trois terminaisons sont possibles : une boule, une flèche ou rien. La boule signifie que l'interaction est bénéfique pour l'espèce. À

l'opposé, la flèche signifie que l'interaction est négative pour l'espèce. Lorsque la terminaison ne comporte rien, ceci signifie que l'interaction n'a aucune influence sur l'espèce. Dans l'exemple présent sur la Figure 20, l'*espèce 1* influence négativement l'*espèce 2*, car la première se nourrit de la seconde (prédation). À l'opposé, l'*espèce 2* influence positivement l'*espèce 1* en la nourrissant. Un cas spécial est prévu pour l'interaction de type « compétition ». La compétition implique la dispute de deux espèces au sujet d'une ressource épuisable. Donc, la ressource en cause est incluse dans la représentation au dessus de la flèche bidirectionnelle. Ceci permet de voir rapidement que l'*espèce 1* et l'*espèce 3* ont un comportement néfaste l'une pour l'autre, car elles se disputent la *ressource 1*.

Le vocabulaire, la structure statique et la structure dynamique ont le potentiel de permettre de comprendre graphiquement le comportement d'un système complexe. Avant de pouvoir tester cette affirmation le modèle conceptuel doit être transposé en langage informatique. Une architecture logicielle représentant la structure et le comportement du modèle conceptuel sera élaborée à cette fin. Cette architecture est présentée à la section suivante.

Architecture du Framework implantant le modèle conceptuel

Cette section décrit l'architecture du Framework implantant le modèle conceptuel présenté aux sections précédentes. De par ses emprunts au domaine de la biologie, le Framework a été baptisé « Écosystème ». Cette architecture vise à décrire « *d'une manière symbolique et schématique les différents éléments d'un ou de plusieurs systèmes informatiques, leurs interrelations et leurs interactions* » [72]. Les éléments et les concepts présentés à la section « Modèle conceptuel » sont au cœur de l'architecture proposée. Premièrement, les mécanismes et les éléments nécessaires à la création d'un écosystème seront présentés. Deuxièmement, la structure logicielle reflétant le concept d'écosystème sera décrite. Troisièmement, l'aspect dynamique du Framework sera détaillé. Finalement, la programmation orientée objet dans MATLAB® avec laquelle le Framework Écosystème est implanté sera survolée.

Création d'un écosystème

La phase de création consiste en l'instanciation des éléments qui sont présents dans l'écosystème à l'étude. C'est à ce niveau que l'écosystème, les ressources, les espèces, les organismes et les propriétés des organismes sont créés d'un point de vue logiciel. Cette section présente la structure logicielle des ces éléments.

Le premier élément présenté est l'écosystème lui-même. La classe définissant l'écosystème est présentée à la Figure 21.

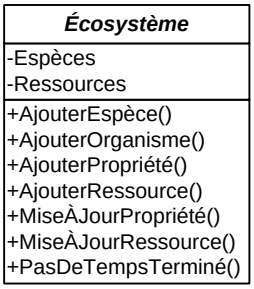


Figure 21 Classe Écosystème

Cette classe abstraite représente le point d'accès au Framework. Tout d'abord, c'est par les fonctions de cette classe qu'il est possible d'ajouter différents éléments à l'écosystème comme les ressources, les espèces et les organismes ainsi que leurs propriétés. Ensuite, la mise à jour, en cours de simulation, des propriétés et des ressources se fait aussi par l'utilisation de ces fonctions. Finalement, la fonction *PasDeTempsTerminé* doit être redéfinie par l'utilisateur. C'est à l'intérieur de cette dernière qu'il est possible d'accéder à la valeur des propriétés des différents organismes et d'appeler les différents algorithmes de détection d'interaction.

La classe Écosystème peut être vue comme le monteur dans le patron de conception logicielle *monteur* [73] comme l'illustre la Figure 22 dans le cas de la création d'une ressource.

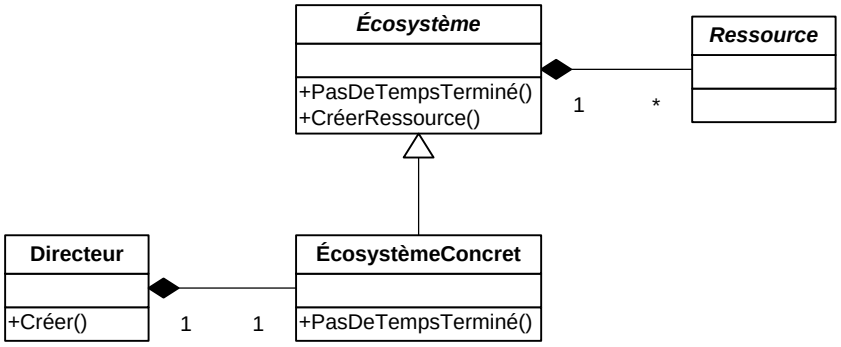


Figure 22 Création d'une ressource en utilisant Écosystème

Le patron logiciel a été adapté pour les besoins actuels. En effet, le produit créé deviendra un attribut de l'Écosystème et ne sera accessible que dans la fonction *PasDeTempsTerminé* comme décrit précédemment.

De plus, le *Directeur* représente la partie de l'engin de simulation qui utilise le Framework par le biais d'une classe héritant d'*Écosystème*. Donc, lorsque le *Directeur* désire créer un élément de l'*Écosystème* comme une *Ressource*, il fait la requête par le biais des fonctions de l'*Écosystème*.
Le premier élément qui peut être créé par l'*Écosystème* est une *Ressource*. La Figure 23 montre la classe la représentant.

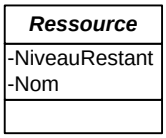


Figure 23 Classe Ressource

Cette classe est abstraite ce qui permet l'insertion de nouvelles ressources au Framework par un utilisateur externe.

L'Écosystème permet aussi la création des espèces (Figure 24).

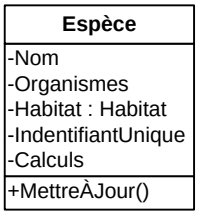


Figure 24 Classe Espèce

Lors de la création de l'espèce, l'utilisateur n'a qu'à se soucier du nom qu'il veut lui donner. En effet, les autres attributs seront associés à l'espèce par la création des *Organismes* et de leur *Propriétés*.

Pour qu'une *Espèce* existe, elle doit être formée d'au moins un *Organisme*. Donc, l'*Écosystème* permet aussi de créer un *Organisme* et de l'associer à une *Espèce*. La Figure 25 présente la classe représentant un *Organisme*.

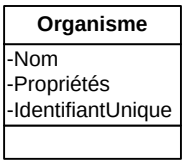


Figure 25 Classe Organisme

Un *Organisme* regroupe un ensemble de *Propriétés*. La création d'un *Organisme* passe donc par la création des propriétés qui en font un organisme indépendant.

Finalement, pour compléter la présentation des éléments d'un *Écosystème*, la *Propriété* est présentée à la Figure 26.

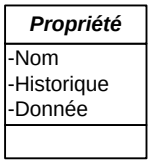


Figure 26 Classe Propriété

La *Propriété* renferme la donnée brute ainsi que son historique à travers le temps. L'utilisateur peut y accéder comme mentionné précédemment par le biais de l'*Écosystème* à l'intérieur de la fonction *PasDeTempsTerminé*.

La Figure 27 présente un exemple de création d'un *Écosystème*.

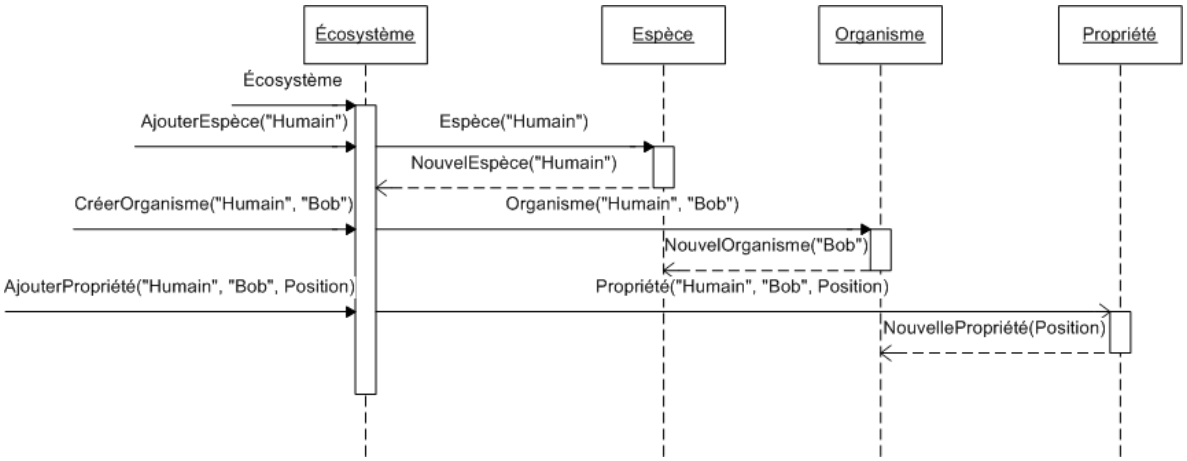


Figure 27 Diagramme séquentiel de la création d'un *Écosystème*

Dans le cas présenté à la Figure 27, l'utilisateur crée un *Écosystème* contenant une *Espèce*, un *Organisme* et une *propriété*. Bien entendu, ce type de scénario ne sert qu'à montrer le processus de création de chaque élément de l'*Écosystème*, car aucune interaction ne peut être détectée avec la présence d'une seule *Espèce*. La première étape consiste à créer l'*Écosystème* par l'utilisation du constructeur (*Écosystème*). Une fois l'*Écosystème* créé, l'utilisateur y ajoute des *Espèces* (*AjouterEspèce*). Dans le cas présent, l'espèce *Humain* est créée. Par la suite, les *Organismes* sont instanciés (*CréerOrganisme*). Au même moment, ils sont associés à une *Espèce* en particulier. Dans l'exemple, *Bob* est un *Organisme* qui est ajouté à l'espèce *Humain*. Finalement, les *Propriétés* sont créées (*AjouterPropriété*) et assignées aux *Organismes*. Dans le cas présent, *Bob* qui est un *Humain* à un *Position*.
Chaque élément de l'écosystème a été couvert de façon indépendante. Par contre, c'est leur regroupement qui fait la force du Framework. La structure logicielle du Framework est présentée dans la section suivante.

Structure d'un écosystème

Chaque élément de l'*Écosystème* renferme des caractéristiques propres qui peuvent être utiles à l'utilisateur. Par contre, c'est l'utilisation de l'ensemble de ces caractéristiques individuelles qui permettent de déceler des comportements complexes. Dans le cas présent, ces comportements sont représentés sous forme d'interactions. Ceci implique qu'une bonne compréhension la structure statique du Framework composée des éléments présentés à la section précédente est essentielle. Cette section présente la structure statique de l'*Écosystème*. Il est à noter qu'une attention toute particulière est consacrée aux propriétés et aux interactions.

La structure statique [74] est une représentation orientée-objet du modèle conceptuel. Ce qui signifie que le modèle conceptuel décrit à la section « Modèle conceptuel » est représenté sous forme de classes et des relations entre celles-ci. Il est à noter que la structure statique n'illustre que certains indices du comportement dynamique du système, comme l'énumération de fonctions. Les détails du comportement dynamique du système sont couverts par d'autres diagrammes présentés à la section suivante (« Structure dynamique d'un écosystème »).

La vue statique d'ensemble de l'*Écosystème* est présentée à la Figure 28.

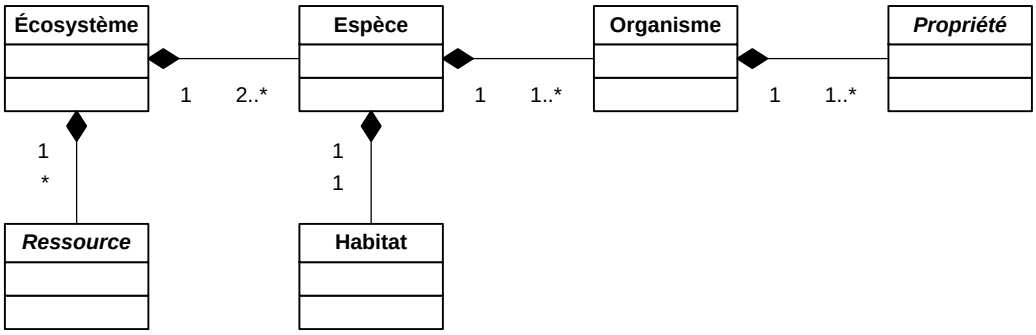


Figure 28 Structure d'un écosystème (diagramme de classes)

Sur la Figure 28 on peut voir tous les éléments principaux de l'*Écosystème* décris dans le modèle conceptuel. La structure consiste tout simplement en une chaîne d'agrégations par composition [75], ce qui signifie que chaque élément est composé d'un sous-ensemble d'éléments de façon telle que lorsqu'un élément est détruit, tout les éléments qui le composent le sont aussi.

L'*Écosystème* est composé au premier niveau d'*Espèces* et de *Ressources*. Il se peut que pour certains scénarios de simulation, il n'y ait pas de *Ressources*. Par contre, un *Écosystème* doit inclure au moins 2 *Espèces*. Théoriquement, un *Écosystème* pourrait être composé d'une seule espèce, mais, dans ce cas, le Framework perdrait son utilité car, pour qu'il y ait interaction, au moins deux *Espèces* doivent coexister dans l'écosystème.

Une *Espèce* est composée d'un *Organisme* ou plus et d'un *Habitat*. Sans les *Organismes*, l'*Espèces* n'existe pas. Une *Espèce* peut être réduite à un seul *Organisme* dans certains cas d'exception, par exemple dans des cas où la population est très petite, ou dans les cas où l'utilisateur désire se concentrer sur l'interaction d'un seul *Organisme* avec les autres ou dans le cas où les interactions entre *Espèce* résultent en ce qu'un seul *Organisme* de l'*Espèce* survit. En ce qui à trait à l'*Habitat*, sa configuration est calculée à chaque pas de temps en fonction de la position et du rayon d'action des *Organismes* qui le composent. Ce sujet sera abordé à la section « Calcul de l'*Habitat* », car l'*Habitat* est essentiel à la détection d'interactions et il fait partie des composants dynamiques de l'écosystème.

Ce qui fait qu'un *Organisme* est unique par rapport aux autres membres de son *Espèce* est la valeur de ses *Propriétés* (i.e. son « état » en langage de génie logiciel). Ce sont ces dernières qui le définissent et en font ce qu'il est. Ceci implique qu'un *Organisme* sans *Propriétés* n'existe pas.

Les *Propriétés* sont à la base du comportement dynamique du système, car ce sont ces dernières qui sont analysées pour détecter les interactions. Le diagramme de classes présentant la structure des *Propriétés* est représentée à la Figure 29.

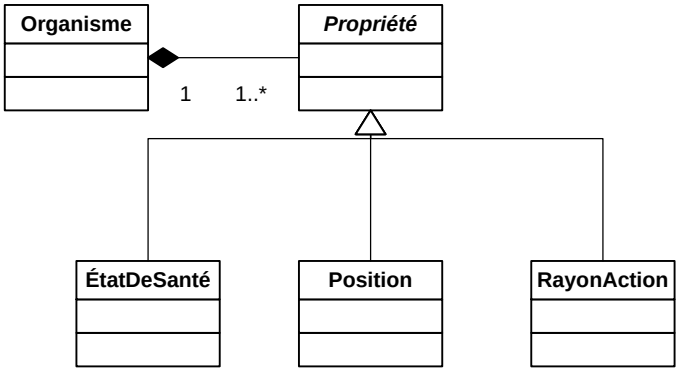


Figure 29 Structure des propriétés (diagramme de classes)

Les *Propriétés* de base qui seront utilisées lors de la détection d'interactions sont montrées dans le diagramme soit : l'*ÉtatDeSanté*, la *Position* et le *RayonAction*. Elles sont toutes utiles à la détection d'interactions qui sera présentée à la section « Algorithme de détection d'interactions ». Lorsqu'un utilisateur désire ajouter des analyses qui nécessitent des *Propriétés* additionnelles, il n'a qu'à les ajouter à la hiérarchie de classes.

Les *Interactions*, qui représentent le dernier élément de la structure statique de l'Écosystème, sont présentées à la Figure 30.

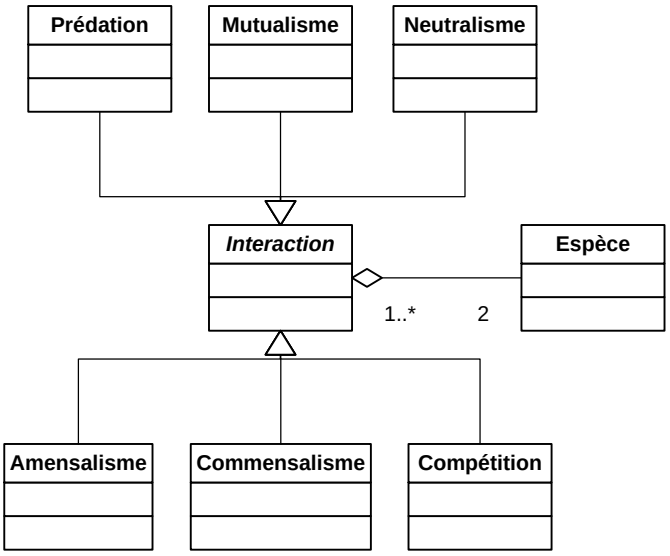


Figure 30 Structure des interactions (diagramme de classes)

La Figure 30 montre que chaque interaction hérite de la classe abstraite *Interaction*. Cette dernière renferme les propriétés communes à chaque type d'interaction. Une Interaction est composée dans tous les cas de seulement deux espèces. Une même espèce peut participer à plusieurs interactions. Par contre, deux espèces ne peuvent pas avoir plus d'un type d'interaction. En effet, une espèce peut être soit neutre, bénéfique ou nuisible à une autre tel que défini au Tableau 1.

La structure statique a été couverte dans cette section et les éléments permettant de créer un *Écosystème* ont été présentés. La prochaine section montre de quelle façon ces éléments sont impliqués dans le comportement dynamique du Framework.

Le comportement d'un écosystème

La structure statique d'un *Écosystème* présentée à la section précédente permet de décrire la façon dont tous les éléments sont agencés entre eux. Par contre, outre la liste des fonctions membres de chacune des classes, le comportement dynamique est absent de la structure statique. Cette section présente le comportement dynamique qui décrit l'évolution du système. Tout d'abord, la fonction *PasDeTempsTerminé* sera présentée. Ensuite, une partie concernant des calculs préliminaires à la détection d'interactions sera introduite. Finalement, les approches d'analyse en cours de simulation du scénario seront présentées.

L'accès au Framework se fait par le biais de la fonction *PasDeTempsTerminé*. En effet, lorsqu'un pas de temps de simulation se termine, l'engin de simulation appelle la fonction *PasDeTempsTerminé* de la classe *Écosystème* (Figure 21). C'est à l'intérieur de cette dernière que les données intrinsèques (propriétés, habitat…) de l'*Écosystème* peuvent être accédées. Ceci permet d'appliquer différents algorithmes de calcul et d'analyse à l'intérieur de cette fonction. La Figure 31 présente la séquence de traitements effectués dans la fonction.

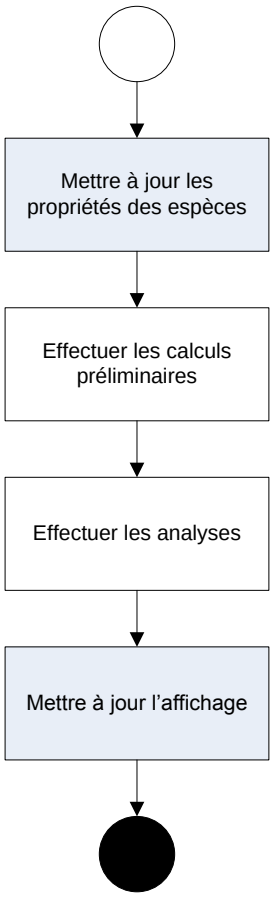


Figure 31 Séquence des traitements pour la fonction-membre *PasDeTempsTerminé* de la classe Écosystème

Les étapes da la fonction *PasDeTempsTerminé* présentée à la Figure 31 sont détaillées à la Figure 32. En effet, la Figure 32 décrit la séquence d'appels de fonction aux objets instanciés à l'intérieur du Framework.

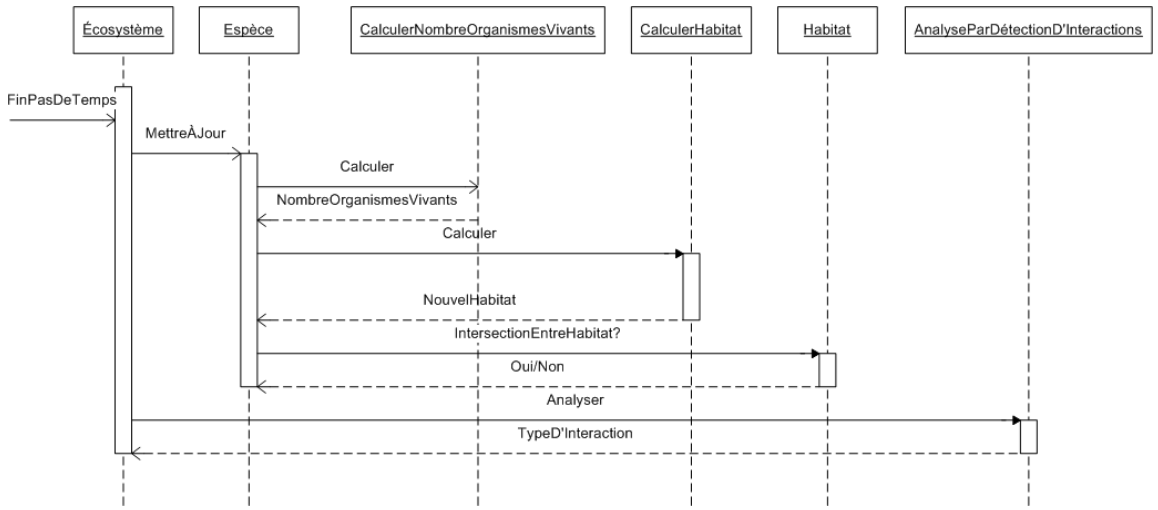


Figure 32 Diagramme séquentiel de *PasDeTempsTerminé*

Le premier traitement du système, lorsque la fonction *PasDeTempsTerminé* est appelée, consiste à synchroniser la valeur des *Propriétés* des *Organismes* avec les données de l'engin de simulation (*MettreÀJour*). C'est aussi lors de cette étape que l'*Écosystème* enregistre la valeur précédente des propriétés modifiées dans l'historique. Une fois l'*Écosystème* mis à jour, le traitement suivant consiste à effectuer des calculs préparatoires. Ces calculs sont la phase préliminaire à la détection d'interactions (*Calculer*). Ils permettent d'obtenir de l'information supplémentaire sur les variations de la valeur des *Propriétés* présentes dans l'*Écosystème*. La Figure 33 montre le diagramme de classe des calculs préliminaires pouvant être effectués.

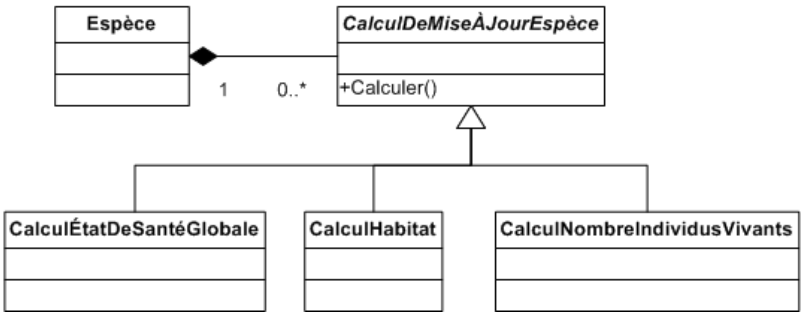


Figure 33 Calculs préparatoires (diagramme classes)

Chaque calcul concret hérite de la classe abstraite *CalculDeMiseÀJourEspèce*. Cette dernière renferme la fonction *Calculer* qui doit être redéfinie dans les sous-classes. De cette façon, toutes les classes permettent

de faire des calculs spécifiques visant à obtenir de l'information supplémentaire à partir de la valeur des *Propriétés* des *Organismes* tout en respectant une structure précise. La Figure 33 présente les calculs inclus dans le Framework et qui seront utiles pour la détection d'interactions. Par contre, il et possible pour un utilisateur d'ajouter un algorithme de calcul en définissant une nouvelle classe qui hérite de la classe *CalculDeMiseÀJourEspèce*. Il est à noter que les calculs se font au niveau de l'*Espèce* et non au niveau de l'*Écosystème* ou des *Organismes*. La raison justifiant cette approche est que le Framework tente d'offrir une compréhension d'ensemble du système complexe à l'étude. Par conséquence, les calculs ne peuvent pas être effectués au niveau de l'*Organisme*, car l'accès aux *Propriétés* d'autres *Organismes* ne serait pas possible de cette façon. De plus, les calculs ne se font pas non plus au niveau de l'*Écosystème* car, à l'opposé de l'*Organisme*, toutes les données sont accessibles ce qui pourrait porter à confusion à cause du niveau de complexité rencontré dans l'*Écosystème*.

Une fois les différents *Calculs* préliminaires effectués, le système entre dans l'étape de traitement qui consiste à effectuer l'analyse de *Propriétés* et des nouvelles valeurs calculées par les algorithmes de traitements préliminaires afin de déceler l'information pertinente à l'analyse du scénario simulé (*Analyser* Figure 32). La détection d'interactions sera la seule développée et présentée dans le cadre des travaux de recherche. Par contre, le Framework a été conçu de façon à ce qu'il soit facile d'ajouter de nouveaux algorithmes d'analyse. Il suffit au programmeur d'ajouter une nouvelle classe renfermant les calculs et qui hérite de la classe *AnalyseInstantannée*. La Figure 34 présent la hiérarchie proposée.

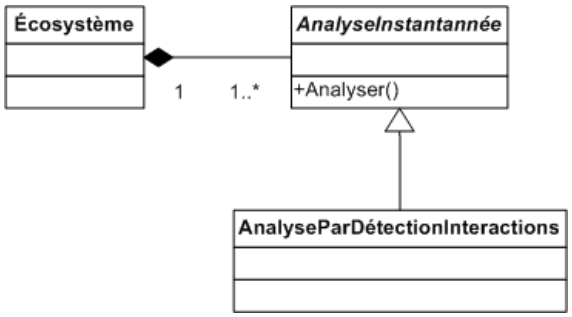


Figure 34 Algorithmes d'analyse (diagramme de classes)

Les diagrammes présentés dans cette section sont de haut niveau. Ils offrent un aperçu de la structure et du fonctionnement du système à développer. En pratique, les caractéristiques du langage choisi pour l'implantation du Framework peuvent influencer l'architecture de plus bas niveau. La section suivante présente le langage qui sera utilisé pour le développement du Framework Écosystème.

Programmation orientée objet avec MATLAB

L'étape suivant la réalisation du modèle conceptuel et la création de l'architecture consiste en l'implantation de celle-ci. Le code obtenu concrétise les différents concepts présentés à la section « Modèle conceptuel ». Cette section présentera la programmation orientée objets (POO) sous le logiciel MATLAB [76] qui est utilisée dans le cadre des travaux. MATLAB permet la POO depuis la version R2008a. Tout d'abord, une brève comparaison entre MATLAB et les autres langages orientés objets (LOO) sera présentée. Ensuite, l'organisation et la définition de classes sous MATLAB sera abordée. Finalement, les concepts de *handle* et celui *value* seront présentés. Cette section s'inspire de certains passages de la documentation de MATLAB et nécessite une connaissance de base de la POO.

La POO de MATLAB diffère des autres langages comme le C++ et le Java de plusieurs façons. La première différence est qu'il est possible d'utiliser les propriétés d'une classe comme interface public qui est séparée de l'implantation de la gestion des données. Ceci est possible car les fonctions d'assignation (*set*) et de lecture (*get*) de chaque propriété peuvent être définies et exécutées de façon automatique. Voici un exemple de ce concept :

```
monobj.Couleur = 'Blanc';
```

Dans cet exemple, la chaîne *'Blanc'* est assignée à la propriété *Couleur* de l'objet *monobj*. Avant d'effectuer toute assignation de valeur, *monobj* exécute une méthode appelée *set.Couleur* (en assumant que la classe *monobj* possède une telle méthode) qui permet d'effectuer les opérations nécessaires. Il est à noter qu'il est possible de contrôler l'accès à chaque propriété par l'assignation d'un attribut. L'accès à une propriété peut soit être *public*, *privé* ou *protégé*.

La seconde différence est que pour la plupart des autres LOO, le paramètre de l'objet à une méthode est toujours implicite. Dans MATLAB, les objets sont des paramètres explicites aux méthodes qui agissent sur eux. Voici la façon dont le paramètre objet d'une méthode est utilisé lors de l'appel d'une méthode de la classe représentant l'objet :

```
d1 = ajouter(d,3);
d1 = d.ajouter(3);
```

Les deux commandes ci-haut ont le même effet. En effet, dans les deux cas, la méthode *ajouter* est appelée sur l'objet *d*. Cette dernière peut servir à incrémenter une propriété de la classe par exemple. Il est à noter que

la méthode doit retourner l'objet avec la propriété modifiée. En effet, les objets dans MATLAB sont passés par valeur à une méthode et non par référence, ce qui signifie qu'une copie complète de l'objet est passée à la méthode et c'est ce dernier qui est modifié. Donc, dans l'exemple, *d1* sera une copie de *d* avec l'incrément de 3 appliqué.

Une autre différence concerne l'appel à une opération d'une superclasse. Voici la façon dont l'appel à une fonction d'une superclasse en C++, Java et MATLAB :

- C++ : `superclass::method`
- Java : `superclass.method`
- MATLAB : `method@superclass`.

La quatrième différence majeure est qu'il n'y a pas d'équivalence en MATLAB pour les *templates* en C++ [77] ou le *generic* de Java [78]. Par contre, dû au fait que MATLAB est typé de façon faible [79], il est possible d'écrire des fonctions ou des classes qui fonctionnent avec différents types de données.

La dernière différence concerne la surcharge de fonction, car MATLAB n'offre pas cette fonctionnalité.

Les différences techniques entre MATLAB et les autres LOO ont été abordées. Une fois ces différences assimilées, le programmeur peut maintenant se lancer dans la création de nouvelles classes. La définition et l'organisation de classes sous MATLAB seront présentées dans les prochains paragraphes.

Tout d'abord, avant d'entreprendre l'écriture du code, il est nécessaire de comprendre comment une classe est définie sous MATLAB. La définition d'une classe est représentée par un bloc de code délimité par le mot-clé *classdef* au début et le mot-clé *end* à la fin. La définition de la classe est incluse dans un fichier et ce dernier ne peut pas contenir plus d'une définition de classe. La Figure 35 présente un exemple de définition de classe. Cet exemple est tiré de la documentation de MATLAB.

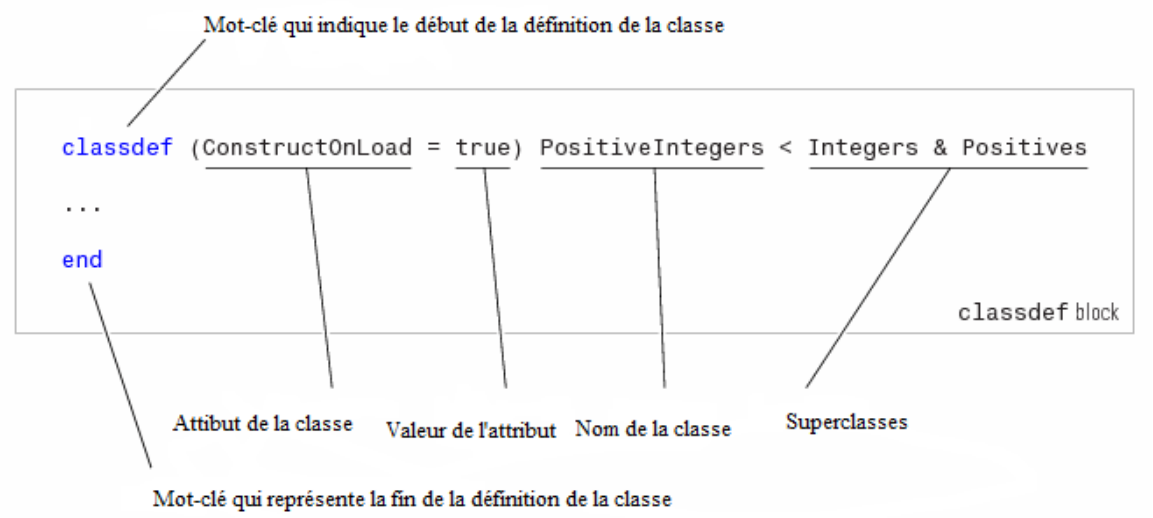


Figure 35 Exemple de définition d'une classe sous MATLAB

La Figure 35 montre la définition de la classe *PositiveIntegers*. Cette dernière hérite de deux superclasses soit *Integers* et *Positives*. Les attributs permettent de changer le comportement de la classe. Pour plus d'informations sur les attributs se référer à la documentation de MATLAB.

Une classe est divisée en deux parties principales. La première renferme toutes les propriétés de la classe. La seconde partie renferme les méthodes associées à la classe. La Figure 36 présente un exemple de classe nommée *class_name*.

```
classdef class_name
   properties
      PropertyName % No default value assigned
      PropertyName = 'some text';
      PropertyName = sin(pi/12); % Expression returns default value
   end
end
```

Figure 36 Définition des propriétés d'une classe MATLAB

La Figure 36 montre les trois façons dont une propriété peut être définie dans une classe MATLAB. La première déclaration de la propriété *PropertyName* montre qu'il est possible de créer une propriété sans y assigner de valeur par défaut. La seconde déclaration présente le cas où une valeur statique est assignée par défaut à la propriété. Dans ce cas, *PropertyName* prend la valeur de '*some text*'. La dernière méthode

d'assignation d'une valeur par défaut montre qu'il est aussi possible d'utiliser une expression dont le résultat sera la valeur par défaut.

La seconde partie qui compose une classe est la définition de ses méthodes. La Figure 37 montre un exemple.

```
classdef ClassName
   methods
      function obj = ClassName(arg1,arg2,...)
         obj.Prop1 = arg1;
         ...
      end
      function normal_method(obj,arg1,...)
         ...
      end
   end
   methods (Static = true)
      function static_method(arg1,...)
         ...
      end
   end
end
```

Figure 37 Définition des méthodes pour une classe MATLAB

La Figure 37 montre tout d'abord la façon dont le constructeur de la classe est défini (méthode *ClassName*). La seconde méthode représente une méthode standard pour une classe sous MATLAB. Il est à noter que l'objet implicite passé en argument à la méthode *normal_method* est représenté par le premier argument dont le nom est *obj*. Finalement, la troisième méthode représente le cas d'une méthode statique. Dans ce cas particulier, il n'est pas nécessaire d'instancier un objet de la classe pour avoir accès à la méthode. C'est pour cette raison que le premier argument de *static_method* ne représente pas un objet de la classe.

Comme le démontre la dernière méthode de la Figure 37, il est possible d'associer un attribut à certains éléments d'une classe. Ces attributs ont pour objectif de changer le comportement de la classe ou des ses éléments (propriétés, méthodes et événements). Ils sont très utiles, car ils permettent de changer facilement le comportement des objets d'une classe sans compliquer le code inutilement.

```
properties (SetAccess = private)
   ScreenSize = getScreenSize;
end
```

Figure 38 Attribut MATLAB pour une propriété

La Figure 38 présente un exemple d'attribut pour une propriété. Dans ce cas, la propriété *ScreenSize* est *private*. Chaque élément d'une classe peut donc avoir ses propres attributs. Pour connaître les types d'attributs disponibles pour chaque élément de la classe, le lecteur peut se référer à la documentation de MATLAB [76].

Un concept important à aborder est celui des classes disponibles dans MATLAB. Deux types de classes s'offrent à l'utilisateur : une classe de type *handle* et une classe de type *value*. Le choix du type se fait en fonction du comportement désiré pour l'instance de la classe et quelles options l'utilisateur désire utiliser. D'une part, une classe de type *value* implique qu'il n'y a aucune référence sur l'objet appartenant à cette classe. En effet, ce type est toujours associé à un « workspace » MATLAB ou à une variable temporaire. Par le fait même, l'objet de type *value* cesse d'exister lorsqu'il n'est plus utilisé par le logiciel ou lorsqu'il est éliminé de la mémoire. Ce type de classe est donc utile lorsque l'utilisateur désire faire une copie de l'objet à chaque fois que ce dernier est assigné à une autre variable ou lorsqu'il est passé en paramètre à une méthode. D'autre part, un objet d'une classe de type *handle* réfère à un objet de la classe. En effet, une classe de type *handle* utilise des descripteurs (« handles » en anglais) qui permettent d'identifier une instance de la classe. Ainsi, lorsqu'une copie de ce type de classe est effectuée, le descripteur est copié mais pas les données emmagasinées dans l'objet. La copie réfère donc aux mêmes données que l'objet original. Donc, lorsque la valeur de données change, tous les objets copiés seront affectés par le changement. La Figure 39 présente la façon dont une nouvelle classe (*myClass*) hérite de la classe *handle*.

```
classdef myClass < handle
    ...
end
```

Figure 39 Héritage de la classe *handle*

Cette section a présenté la POO sous MATLAB qui est l'outil de développement exploité pour l'implantation du Framework *Écosystème*. Les différents aspects abordés précédemment présentent les particularités qui ne sont pas implicites aux autres langages de programmation OO comme le C++ et le Java. Le chapitre suivant présente les algorithmes de détection d'interaction qui seront développés sous MATLAB.

Chapitre 3 : Détection d'interactions

Tel que présenté au chapitre précédent, les interactions sont au cœur même de la compréhension des systèmes complexes simulés. En effet, ces systèmes étant non-déterministes, une analyse séparée de leurs parties constituantes ne permet pas de comprendre le comportement global du système. Dans cette thèse nous avançons que l'analyse des interactions entre ces parties est une façon potentiellement plus efficace de comprendre l'évolution d'un système complexe. Ce chapitre présente les étapes requises pour la détection et l'identification des interactions à l'intérieur du Framework. D'une part, les calculs préliminaires qui permettent d'effectuer une synthèse des données brutes seront présentés (« Effectuer les calculs préliminaires » sur la Figure 31). D'autre part, l'algorithme d'identification des interactions utilisant l'information obtenue par le biais des calculs préliminaires sera présenté par la suite (« Effectuer les analyses » sur la Figure 31).

Calculs préliminaires

Les composants du Framework permettent l'organisation et le suivi d'une simulation visuelle interactive tel que présenté à la section « Architecture du Framework implantant le modèle conceptuel ». Par contre, les données brutes qui sont synchronisées entre l'engin de simulation et le Framework doivent être manipulées afin de permettre l'extraction d'information supplémentaire. Des algorithmes de calculs préliminaires sont donc utilisés pour générer de nouvelles propriétés appelées « propriétés calculées ». L'organigramme de la Figure 40 présente les étapes relatives aux calculs préliminaires.

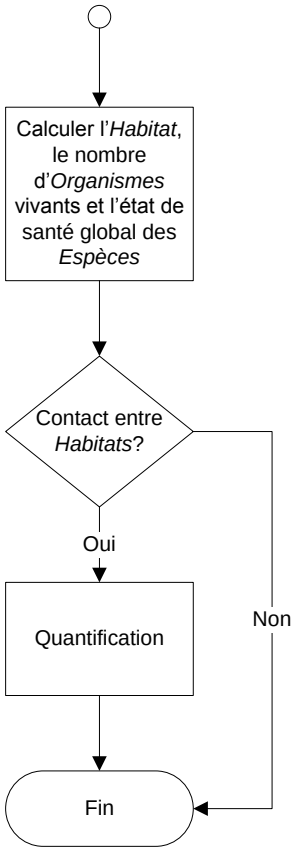


Figure 40 Organigramme pour les calculs préliminaires

Les prochains paragraphes vont expliquer en détails les différentes étapes de l'organigramme présenté à la Figure 40. Tout d'abord, le rôle des propriétés calculées sera présenté afin de bien comprendre l'implication de chacune d'entre elles. Ensuite, les algorithmes de manipulation des données brutes qui servent à calculer les propriétés calculées seront présentés. Ces algorithmes ont pour but de calculer le nombre d'*Organismes* vivants par *Espèce*, l'état de santé global de chaque *Espèce* et l'*Habitat* de chaque *Espèce*. Il est à noter que les trois algorithmes sont lancés à chaque pas de temps pour chaque *Espèce* présente dans l'*Écosystème*. Le Framework emmagasine les propriétés calculées dans un historique, ce qui permet à l'algorithme de détection d'interactions (décrit à la section « Algorithme de détection d'interactions ») d'utiliser cette information. Par la suite, la procédure pour détecter le contact entre *Habitats* sera abordée. Finalement, la quantification [80] des données calculées à la première étape sera présentée. Un exemple sera aussi présenté afin d'illustrer clairement le processus de quantification.

Propriétés calculées

Les propriétés calculées ont pour objectif de synthétiser la viabilité [81] d'une *Espèce*. Ensemble, elles permettent de décrire la capacité d'une *Espèce* à survivre et se développer dans un *Écosystème* précis. Elles seront utilisées pour la détection d'interactions. En effet, la détection d'interactions requiert d'être en mesure d'effectuer le suivi de la viabilité d'une *Espèce* dans le temps par l'analyse des propriétés calculées. De cette façon, il sera possible d'utiliser les propriétés calculées afin de détecter l'influence d'une *Espèce* sur chaque autre *Espèce*. Pour ce faire, trois propriétés calculées sont ciblées pour représenter les capacités de survie et d'épanouissement d'une *Espèce*. Le choix de ces propriétés s'inspire des critères de la liste rouge de l'Union internationale pour la conservation de la nature [82]. Cette liste a pour but d'effectuer une estimation du danger d'extinction d'une *Espèce* en se basant sur des critères bien définis, comme la taille de la population, la disparition de son habitat naturel et le nombre d'individus qui ont atteint la maturité. Dans le contexte de cette thèse, ces trois critères peuvent être transposés comme étant le nombre d'*Organismes* vivants au sein d'une *Espèce*, la dimension de l'*Habitat* pour l'*Espèce* et l'état de santé global de l'*Espèce*. Les sections « Calcul du nombre d'*Organismes* vivants », « Calcul de l'état de santé global d'une *Espèce* » et « Calcul de l'*Habitat* » présenteront la façon dont ces quantités sont calculées.

Tout d'abord, la fluctuation du nombre d'*Organismes* vivants est un bon indicatif de la viabilité d'une *Espèce*. Cette fluctuation peut avoir trois tendances. La première tendance est celle concernant une *Espèce* avec un taux de natalité et de mortalité égaux. Ce qui signifie que l'*Espèce* est stable [83]. La seconde tendance est celle concernant une population où le taux de natalité est supérieur au taux de mortalité. Dans ce cas, l'*Espèce* a une bonne viabilité. Par contre, cette tendance ne peut pas continuer à l'infini. En effet, l'*Habitat* d'une *Espèce* a une capacité porteuse [84] qui représente la capacité maximale d'*Organismes* que l'*Habitat* peut supporter. Les *Ressources* vont pour la plupart du temps dicter cette capacité. En effet, elles doivent être en quantité suffisante pour approvisionner convenablement les *Organismes*. Une fois la capacité porteuse atteinte, l'*Espèce* passe en mode stable si aucun élément externe ne vient l'influencer. La dernière tendance englobe le cas où une *Espèce* est en déclin. Si la tendance de cette courbe associée à une *Espèce* ne change pas, l'*Espèce* sera éventuellement en voie d'extinction. Donc, l'analyse de la tendance de la courbe du nombre d'*Organismes* donne un indice sur la viabilité de l'*Espèce*.

Ensuite, la dimension de l'*Habitat* est aussi un facteur à considérer lors de l'analyse de la viabilité d'une *Espèce*. Chaque *Organisme* d'une *Espèce* a besoin d'une certaine dimension de territoire pour se nourrir, s'abriter et se reproduire. La dimension requise dépend des besoins spécifiques de chacune des *Espèces*. La

fluctuation de la dimension de l'*Habitat* peut donc être un bon indicatif de la viabilité d'une *Espèce*. Par exemple, si une *Espèce* voit son *Habitat* diminuer dramatiquement par l'apparition d'une nouvelle *Espèce* dans l'*Écosystème*, la viabilité de la première *Espèce* est touchée. En effet, la diminution de l'*Habitat* va engendrer une plus grande compétition pour les *Ressources* au sein de l'*Espèce*. Donc, une diminution du nombre d'*Organismes* est à prévoir.

Finalement, la fluctuation de l'état de santé des *Organismes* vivants d'une *Espèce* est aussi un bon indicatif de la viabilité. En effet, un *Organisme* ne possédant pas un bon état de santé aura plus de difficultés à se développer, à survivre et à se reproduire comparativement à un *Organisme* en pleine santé. Par exemple, la saison hivernale peut rendre l'approvisionnement en nourriture et la recherche d'abri plus difficile pour certaines *Espèces* ce qui peut affaiblir ses *Organismes*. Les *Organismes* plus faibles devront compétitionner avec des *Organismes* en meilleure forme pour s'approvisionner en nourriture.

Si chaque propriété calculée permet de donner un bon indice sur la viabilité d'une *Espèce*, ne prendre en compte qu'une seule des trois peut ne pas évaluer correctement la viabilité. En effet, dans certains cas, une seule propriété peut indiquer une tendance contraire aux deux autres. C'est pour cette raison qu'il est préférable de combiner les trois indicatifs pour ensuite tirer des conclusions concernant l'*Espèce*. C'est cette approche qui est mise de l'avant dans la détection d'interactions décrite à la section « Algorithme de détection d'interactions ».

Calcul du nombre d'*Organismes* vivants

Le premier indicatif de viabilité d'une Espèce est le nombre d'*Organismes* vivants qu'elle contient. Les *Organismes* vivants dans l'*Écosystème* ont un *ÉtatDeSanté* qui varie dans le temps. Ils peuvent être en très bonne santé (*ÉtatDeSanté* près de 100%) ou ils peuvent être à l'agonie (*ÉtatDeSanté* près de 0%). Ainsi, il est très simple de connaître quels *Organismes* sont vivants en regardant individuellement chaque valeur d'*ÉtatDeSanté*. En effet, si un *Organisme* n'est plus vivant, sont *ÉtatDeSanté* sera de 0%. Donc, l'algorithme consiste tout simplement à vérifier quels *Organismes* ont un *ÉtatDeSanté* égal à 0%. Il est à noter que le Framework garde en mémoire les *Organismes* ayant vécu dans l'*Écosystème*. De cette façon, si un utilisateur désire créer un algorithme qui analyse les *Organismes* décédés, il lui sera possible de le faire.

Calcul de l'état de santé global d'une *Espèce*

Au moment où le Framework connaît quels *Organismes* sont vivants, il peut calculer l'état de santé global pour chaque *Espèce*. L'algorithme consiste tout simplement à prendre la moyenne des *ÉtatDeSanté* de chaque *Organisme* vivant. La valeur résultante du calcul se situera donc entre 0% et 100%. Le Tableau 2 présente un exemple de la valeur d'*ÉtatDeSanté* pour les *Organismes* d'une même *Espèce*.

Tableau 2 Exemple de valeur d'*ÉtatDeSanté* pour le calcul de l'état de santé global d'une *Espèce*

Organisme	*ÉtatDeSanté (%)*
Organisme1	40
Organisme2	100
Organisme3	80
Organisme4	0
Organisme5	22

Dans le cas de l'exemple présenté par le Tableau 2, seul l'Organisme4 est décédé. Donc, le nombre d'*Organismes* encore vivants est de 4 ($n = 4$). Le calcul de l'état de santé global se fait en utilisant l'Équation 1.

Équation 1 Calcul de l'état de santé global

$$[1] \qquad \text{État de santé global} = \frac{1}{n}\sum_{i=1}^{n} \text{É}tatDeSanté_i = \frac{40+100+80+22}{4} = 60.5\%$$

Donc, pour l'exemple présenté, l'état de santé global pour l'*Espèce* sera de 60.5%. Il est à noter que dans le cadre des travaux de la thèse, la moyenne est utilisée pour calculer l'état de santé global (Équation 1). Par contre, le Framework permet de changer cette équation au besoin. En effet, un utilisateur peut développer une équation et l'insérer au Framework par l'ajout d'une classe qui hérite de la classe *CalculNombreIndividusVivants* (Figure 33). Donc, le Framework pourra utiliser la nouvelle équation pour calculer la valeur de l'état de santé global des *Espèces* d'un *Écosystème*.

Calcul de l'*Habitat*

Le calcul de l'*Habitat* est le dernier algorithme de manipulation des données brutes avant la détection d'interactions entre les *Espèces*. Ce calcul consiste à prendre la *Position* et le *RayonAction* des *Organismes*

composant une *Espèce* pour ensuite en approximer un *Habitat*. Dû au fait que l'engin de simulation utilisé est en deux dimensions (voir section « Engin de simulation multi-agents NetLogo »), l'algorithme permettant de calculer l'*Habitat* sera présenté à l'aide d'un repère de coordonnées 2D. Par contre, le Framework supporte autant les coordonnées en trois dimensions que celles en deux dimensions. La Figure 41 présente les différentes étapes nécessaires pour calculer l'*Habitat*.

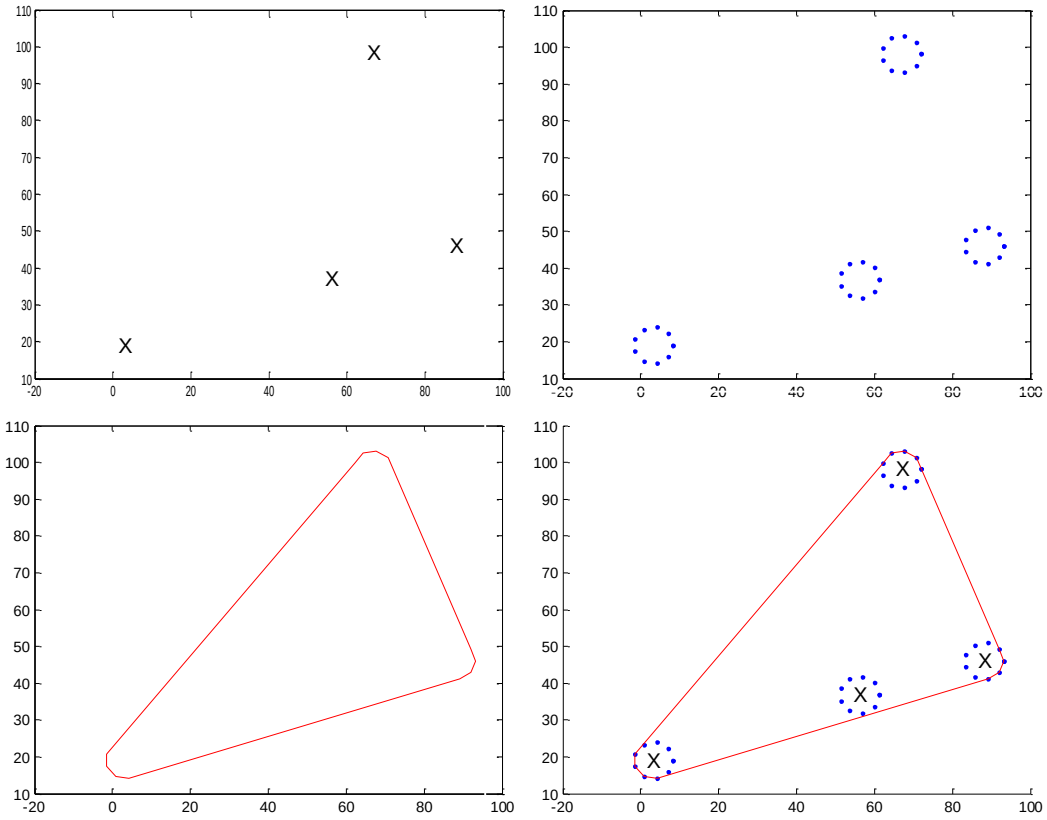


Figure 41 Étapes du calcul de l'*Habitat*

La première étape (haut gauche sur la Figure 41) du calcul de l'*Habitat* consiste tout simplement à obtenir la *Position* (i.e. ses coordonnées dans l'*Habitat*) de chaque individu de l'*Espèce* à analyser. Il est à noter que la synchronisation entre les données du simulateur et du Framework a été effectuée auparavant à l'intérieur de la fonction *PasDeTempsTerminé* tel qu'expliqué à la section « Le comportement d'un écosystème ».

La seconde étape (en haut à droit sur la Figure 41) consiste à créer une zone d'interaction possible pour chaque *Organisme*. Cette dernière prend en compte la *Position* de l'*Organisme* ainsi que son *RayonAction*. Ce qui signifie que cette zone représente la partie de l'*Habitat* où l'*Organisme* en question peut interférer avec d'autres *Organismes*. Donc, lorsqu'un *Organisme* se trouve dans une telle zone, il y a une possibilité d'interaction. La zone d'interaction pour chaque *Organisme* est approximée par un cercle dont le rayon équivaut au *RayonAction* de l'*Organisme* en question et est centrée sur sa *Position*. Ce cercle est approximé par un polygone simple [85]. Le polygone est utilisé, car il peut représenter un cercle et des algorithmes permettent de détecter facilement l'intersection de deux polygones (la fonction MATLAB *inpolygon*). Le choix du nombre de points pour représenter le cercle en forme de polygone est relatif à deux facteurs. Le premier est la résolution désirée pour le cercle. En effet, plus grand le nombre de points, plus précise sera la représentation du cercle. La partie de gauche de la Figure 42 montre ce phénomène. On y voit deux cercles de même rayon mais ayant un nombre de points différents pour le représenter. Le polygone rouge renferme plus de points donc sa représentation est plus près de celle d'un cercle.

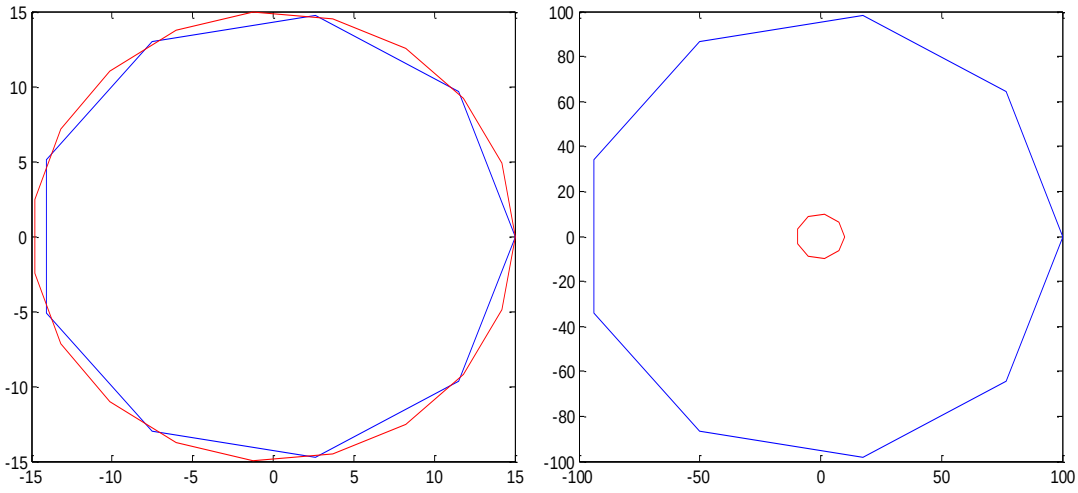


Figure 42 Approximation d'un cercle en polygone

Le second facteur est le rayon du cercle à approximer. En effet, plus le cercle à une grande circonférence plus il faudra de points pour le représenter. La partie de droite de la Figure 42 présente deux polygones ayant le même nombre de points mais de rayons différents. Le constat est que le polygone en rouge s'apparente plus à un cercle que le polygone en bleu. Sur l'exemple de la Figure 41 le polygone est composé de neuf points. Le choix du nombre de points est donc dépendant du scénario et doit être évalué au cas par cas.

La troisième étape (bas gauche sur la Figure 41) consiste à fusionner les zones d'interaction pour chaque *Organisme* d'une *Espèce*, ce qui va représenter l'*Habitat*. Pour ce faire, l'algorithme prend tous les points représentant chaque zone d'interaction calculée à l'étape précédente et en calcule l'enveloppe convexe [86]. La Figure 43 et la Figure 41 (en bas à droite) présentent deux exemples de résultat du calcul de l'enveloppe convexe (l'enveloppe en rouge et les points en bleu) en utilisant la fonction *conhull* de MATLAB [87].

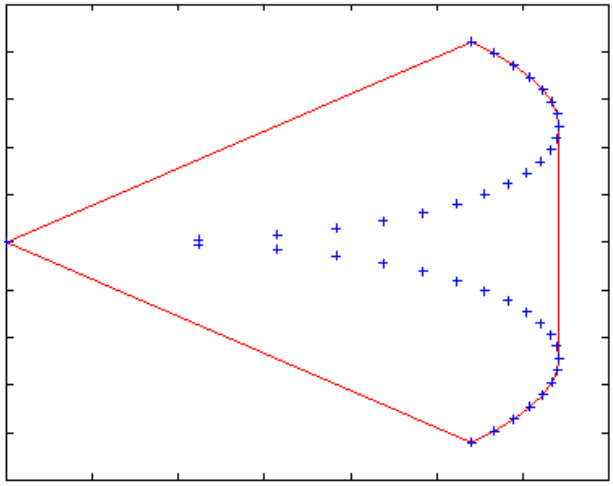


Figure 43 Exemple de résultat de la fonction convhull de MATLAB

Cette méthode est utilisée pour approximer l'*Habitat* qui regroupe l'ensemble des *Organismes* en tenant compte de leur *RayonAction*. Donc, l'ensemble convexe calculé représente le plus petit polygone [88] pouvant englober tout les points calculés à l'étape précédente.

Le calcul de l'*Habitat* sert principalement à détecter le cas de deux *Espèces* dont les *Habitats* sont en recouvrement. Ce qui implique que lorsque deux *Habitats* sont en chevauchement, il y a possibilité d'interaction entre les deux espèces. Dans le cas contraire, il ne peut pas y avoir d'interaction, car le calcul de l'*Habitat* tient compte du *RayonAction* de chaque *Organisme* et que ces derniers n'ont pas d'influence à l'extérieur de leur rayon de portée. Pour vérifier s'il y a un chevauchement d'*Habitats*, la fonction MATLAB *inpolygon* est utilisée. Cette dernière permet de détecter lorsqu'un point d'un polygone se trouve à l'intérieur d'un second. Donc, s'il y a un point d'un *Habitat* se trouvant à l'intérieur d'un autre *Habitat*, les deux *Espèces* impliquées ont la possibilité d'interagir entre elles.

Quantification des données temporelles

Au moment où il y a un premier contact entre deux *Habitats*, le processus de quantification des données temporelles s'enclenche pour les deux *Espèces* impliquées. Il est à noter que le processus de quantification se fait en comparant les valeurs d'amplitude des éléments calculés aux sections « Calcul du nombre d'*Organismes* vivants », « Calcul de l'état de santé global d'une *Espèce* » et « Calcul de l'*Habitat* » lorsqu'il y a contact entre les *Habitats* (exemple dans l'encadré droit sur la Figure 45) et lorsque les *Habitats* ne sont pas en contact (exemple dans l'encadré gauche sur la Figure 45). L'organigramme pour la quantification est présenté à la Figure 44.

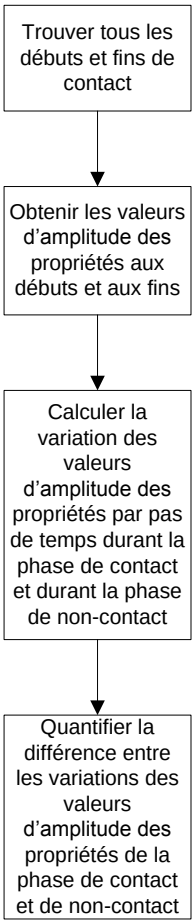


Figure 44 Organigramme pour la quantification des données temporelles

La quantification comporte quatre étapes. La première étape consiste à trouver à quel instant le contact entre deux *Habitats* débute et à quel moment il se termine. Comme expliqué précédemment, la fonction *inpolygon* est utilisée à chaque pas de temps pour détecter cette situation. Lorsque c'est le cas, le Framework en prend

note. Donc, lorsqu'il est temps de connaître le début et la fin d'un contact entre *Habitats*, il suffit d'explorer la liste de moments de contacts. La Figure 45 présente un exemple d'amplitude pour une propriété calculée par pas de temps de simulation. De plus, la figure présente aussi les débuts et les fins de contacts entre les *Habitats* (en pointillé).

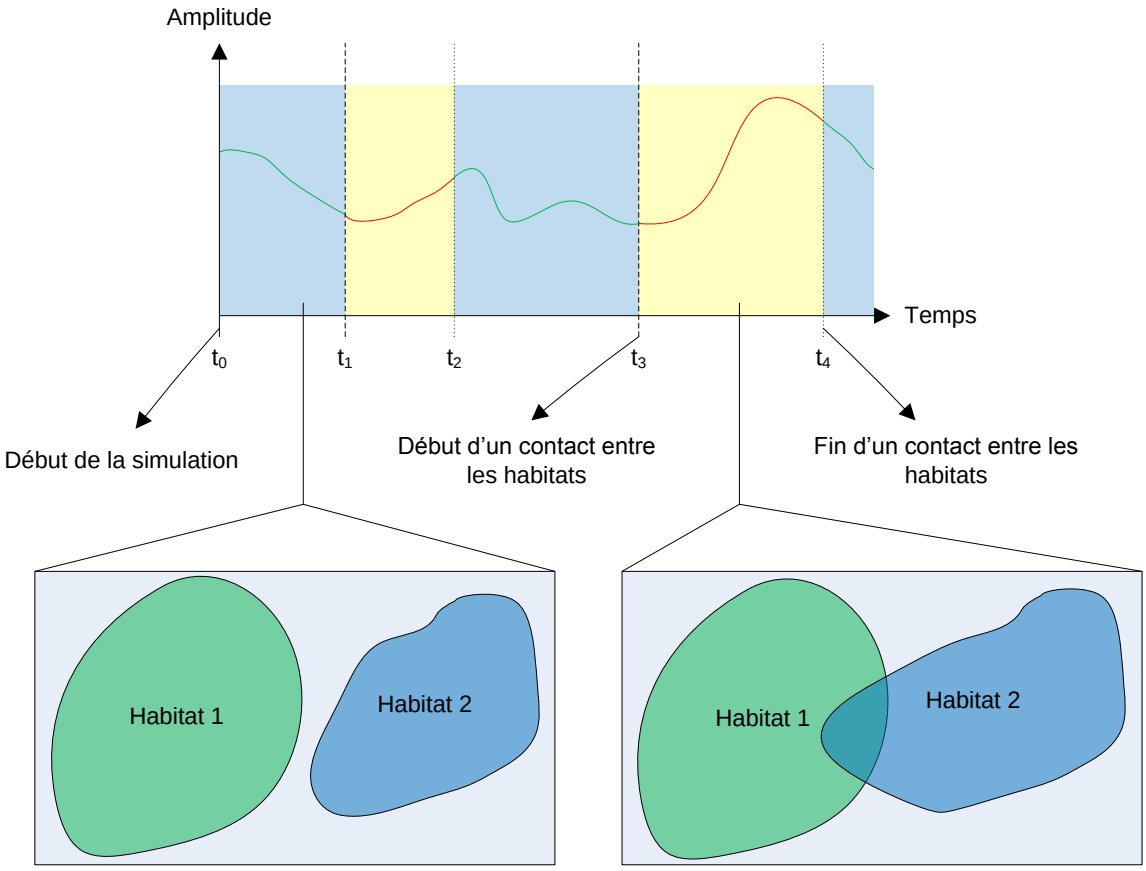


Figure 45 Exemple de variation d'une propriété calculée dans le temps

Sur la Figure 45 on constate deux types de zones temporelles. D'une part, l'ombragé bleu est présent lorsqu'il n'y a pas de contact entre les *Habitats* (non-contact). D'autre part, lorsqu'il y a contact entre les *Habitats* la zone est ombragée en jaune (contact). Donc, au pas de temps t_1 et t_3 se trouvent des débuts de contacts entre les *Habitats* et au pas de temps t_2 et t_4 surviennent des fins de contacts.

La seconde étape consiste à obtenir la valeur d'amplitude des propriétés calculées pour les deux *Espèces* au début des zones temporelles et à la fin de ces dernières. Comme mentionné précédemment, les zones temporelles sont soit les zones où il n'y a pas de contact ou des zones où il y a contact entre les *Habitats*. Sur

la Figure 45 les débuts des zones temporelles de contact se trouvent à t_1 et t_3 et les fins à t_2 et t_4. Pour ce qui est des zones temporelles de non-contact, les débuts se trouvent à t_0, t_{2+1}[4] et t_{4+1} et les fins de contact à t_{1-1} et t_{3-1}. Les éléments analysés dans le cadre du projet sont la dimension de l'*Habitat* qui dépend directement de la *Position* et du *RayonAction* des *Organismes* de l'*Espèce*, le nombre d'*Organismes* vivants pour chacune des espèces en cause et l'*ÉtatDeSanté* global des *Espèces*.

La troisième étape consiste tout d'abord à prendre les valeurs d'amplitude des propriétés calculées pour les deux *Espèces* au début des zones temporelles et à la fin de ces dernières. Ensuite, ces valeurs sont utilisées pour calculer la pente de la dimension de l'*Habitat*, du nombre d'*Organismes* vivants et de l'*ÉtatDeSanté* global pour chaque zone temporelle. Les Équation 2 et Équation 3 présentent ces calculs.

Équation 2 Calcul de la pente pour une zone temporelle de contact entre *Habitats*

$$[2] \qquad \frac{da}{dt} = \frac{\mathrm{a(fin\ contact)} - \mathrm{a(début\ contact)}}{\mathrm{t(fin\ contact)} - \mathrm{t(début\ contact)}}$$

Équation 3 Calcul de la pente pour une zone temporelle de non-contact entre les *Habitats*

$$[3] \qquad \frac{da}{dt} = \frac{\mathrm{a(début\ contact)} - \mathrm{a(fin\ contact)}}{\mathrm{t(début\ contact)} - \mathrm{t(fin\ contact)}}$$

Ceci permet donc de connaître la variation de la valeur d'amplitude à la fin d'une zone temporelle comparativement à sa valeur au début de la zone temporelle.

Il est à noter que le contact entre les *Habitats* peut s'avérer être de longue durée. Pour pallier à cette éventualité, une valeur seuil est utilisée afin fragmenter les zones temporelles de contact entre *Habitats* en plusieurs sous-zones de contact. Cette valeur est assignée par un utilisateur expert du domaine simulé qui sera en mesure de constater lorsque le contact entre les *Habitats* est exceptionnellement long. La Figure 46 présente un exemple de division de la zone temporelle de contact en *Habitats*.

[4] t_{x+1} représente le pas de temps suivant celui à t_x. t_{x-1} signifie le pas de 0 temps précédant t_x.

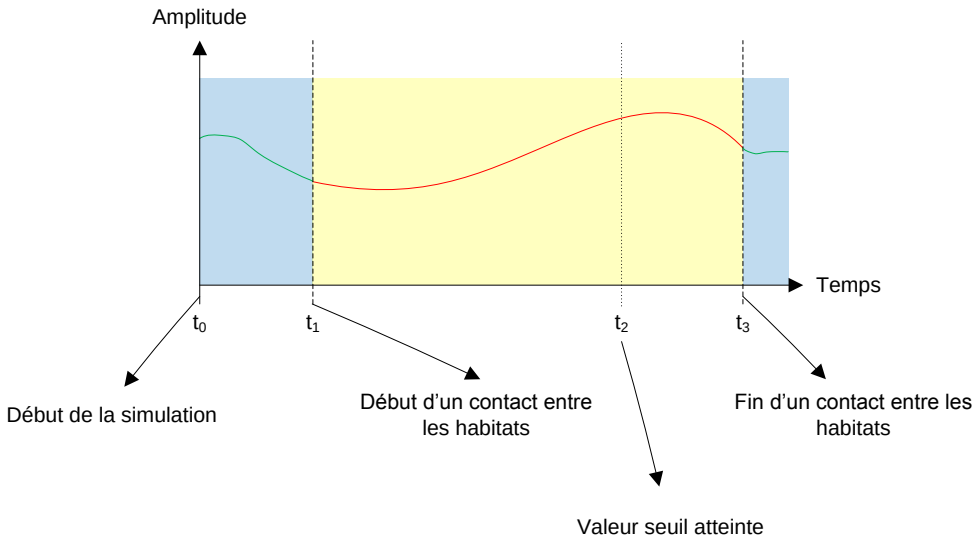


Figure 46 Exemple de fractionnement de la zone temporelle de contact entre *Habitats*

Dans de le cas de l'exemple de la Figure 46, la valeur seuil crée une fin de contact simulée entre les *Habitats* à t_2. En effet, la zone temporelle débutant à t_1 et se terminant à t_3 est divisée en deux parties à t_2. De cette façon, deux zones de contact sont maintenant délimitées et devront être considérées indépendamment dans la suite des calculs.

Une moyenne est ensuite calculée pour toutes les zones temporelles de contact entre les *Habitats* et, aussi, pour les zones où il n'y a pas de contact entre les *Habitats*. Donc, deux valeurs sont disponibles soit la variation par pas de temps de la propriété calculée analysée lors du contact entre *Habitats* ($P_{contact}$) et celle lorsqu'il n'y a pas contact ($P_{non-contact}$).

La dernière étape consiste à prendre les variations calculées à la troisième étape et à comparer celles pour lesquelles y avait contact à celles pour lesquelles il n'y avait pas de contact. Grâce à l'Équation 4 il est possible de comparer les deux valeurs et d'extraire un degré de nocivité (Q) pour un contact entre deux *Habitats*.

Équation 4 Équation de quantification pour les propriétés calculées

$$[4] \qquad Q = \frac{P_{contact} - P_{non-contact}}{ABS(P_{contact}) + ABS(P_{non-contact})}, \; Q = [-1, 1]$$

L'Équation 4 permet de quantifier l'influence d'un contact en lui assignant une valeur entre -1 et 1. Lorsque le contact est nocif pour une Espèce par l'analyse de chaque propriété calculée, la valeur de Q est près de -1. À l'opposé, si la valeur équivaut à 1, le contact entre habitats est bénéfique. Finalement, si la valeur de Q égale 0, il n'y a pas d'influence. Les valeurs de quantification des propriétés calculées seront utilisées dans la suite des calculs.

Exemple du processus de quantification

Pour bien comprendre les quatre étapes de la quantification présentée à la section « Quantification des données temporelles », un exemple est présenté dans cette section. Il est à noter que les étapes de calculs préliminaires préalables à la quantification auront déjà été effectuées. Donc, ces calculs ne seront pas présentés, mais les valeurs résultantes seront utilisées directement. Les quatre étapes de la quantification seront présentées en détail pour un *Écosystème* composé de deux *Espèces* soit des loups et des moutons. La Figure 47 présente l'évolution de propriétés calculées jusqu'au pas de temps courant pour un scénario où des loups se nourrissent de moutons. Les valeurs sont fictives et ne servent qu'à expliquer les différentes étapes de la quantification.

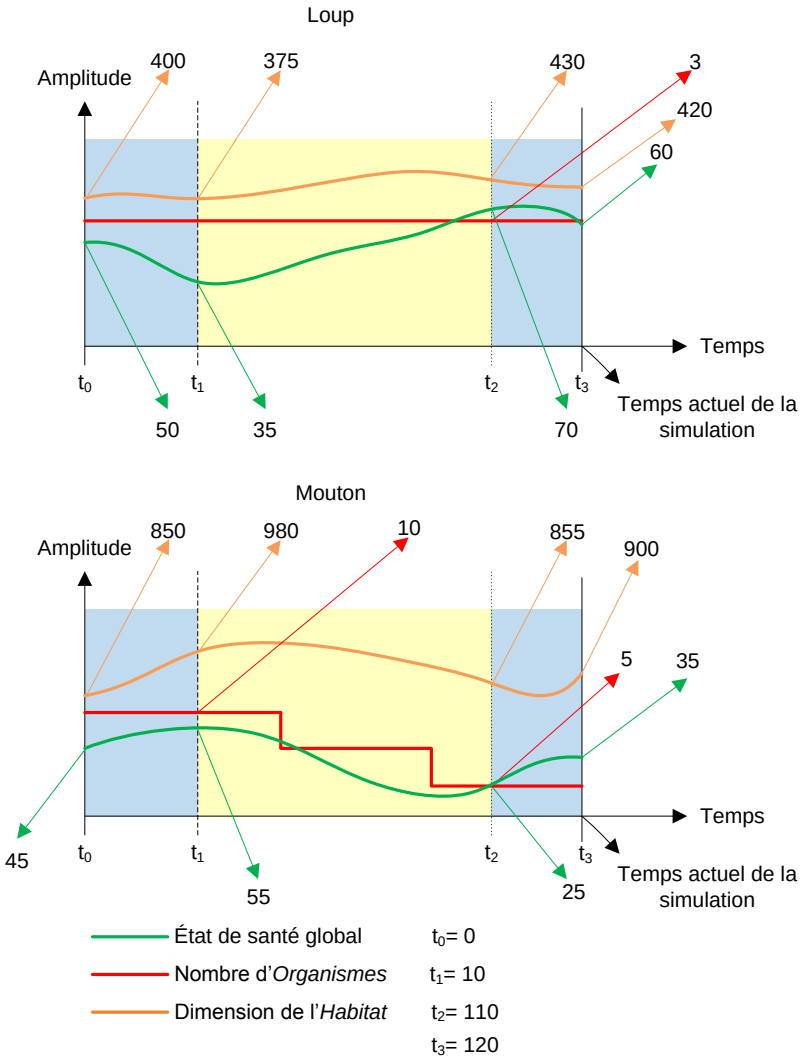


Figure 47 Exemple de propriétés calculées pour un scénario fictif

La première étape consiste à trouver les débuts et les fins pour les trois types de zones temporelles. Le Tableau 3 résume ces instants extraits de la Figure 47.

Tableau 3 Débuts et fins des zones temporelles pour l'exemple de quantification

Contact		Non-contact	
Début	Fin	Début	Fin
t_1	t_2	t_0, t_{2+1}	t_{1-1}, t_3

Cette étape est facilement accomplie dû au fait que le Framework calcule à chaque pas de temps s'il y a un contact entre les *Espèces* de l'*Écosystème* et en prend note dans un liste interne. Il suffit donc de faire la requête au Framework pour connaître les fins et débuts des zones temporelles. Il est à noter que la dernière zone temporelle est bornée par le pas de temps courant de la simulation. Donc, ce dernier sera considéré comme la fin de la zone temporelle qui sera de type de non-contact dans le cas présent. En effet, le dernier événement connu est la fin d'un contact entre les *Habitats* au temps t_2, donc t_3 sera considéré comme faisant partie d'une zone de non-contact.

La deuxième étape consiste à obtenir les valeurs d'amplitudes des propriétés aux positions énumérées par le Tableau 3. Le Tableau 4 présente ces valeurs. Par souci de simplicité, la valeur d'amplitude à t_x est la même qu'à t_{x+1} et t_{x-1}.

Tableau 4 Valeurs d'amplitude des propriétés calculées aux débuts et aux fins des zones temporelles

Temps	Loup			Mouton		
	Nombre	État de santé	Dimension	Nombre	État de santé	Dimension
t_0	3	50	400	10	45	850
t_1	3	35	375	10	55	980
t_2	3	70	430	5	25	855
t_3	3	60	420	5	35	900

Cette étape ne requiert aucun calcul supplémentaire de la part du Framework, car ce dernier effectue les calculs préliminaires à chaque pas de temps et emmagasine l'information dans un historique. Donc, il suffit de faire la requête au Framework pour obtenir les valeurs d'amplitudes aux pas de temps désirés.

La troisième étape consiste à calculer la pente des valeurs d'amplitude présentées au Tableau 4 pour chaque zone temporelle présentée au Tableau 3. Pour ce faire, les Équation 2 et Équation 3 sont utilisées et les résultats sont présentés dans le Tableau 5.

Tableau 5 Résultats de la pente des amplitudes des propriétés pour les zones temporelles

Loup	Mouton
Zone temporelle entre t_0 (0) et t_1 (10) (zone de non-contact entre les *Habitats*)	

Nombre	$\frac{3-3}{10-0}=0$	Nombre	$\frac{10-10}{10-0}=0$
État de santé	$\frac{35-50}{10-0}=-1.5$	État de santé	$\frac{55-45}{10-0}=1$
Dimension	$\frac{375-400}{10-0}=-2.5$	Dimension	$\frac{980-850}{10-0}=13$
Zone temporelle entre t_1 (10) et t_2 (110) (zone temporelle de contact entre les *Habitats*)			
Nombre	$\frac{3-3}{110-10}=0$	Nombre	$\frac{5-10}{110-10}=-0.05$
État de santé	$\frac{70-35}{110-10}=0.35$	État de santé	$\frac{25-55}{110-10}=-0.3$
Dimension	$\frac{430-375}{110-10}=0.55$	Dimension	$\frac{855-980}{110-10}=-1.25$
Zone temporelle entre t_2 (110) et t_3 (120) (zone temporelle de non-contact entre les *Habitats*)			
Nombre	$\frac{3-3}{120-110}=0$	Nombre	$\frac{5-5}{120-110}=0$
État de santé	$\frac{60-70}{120-110}=-1$	État de santé	$\frac{35-25}{120-110}=1$
Dimension	$\frac{420-430}{120-110}=-1$	Dimension	$\frac{900-855}{120-110}=4.5$

Lorsque deux zones temporelles du même type sont détectées, une moyenne est effectuée sur les pentes calculées pour les mêmes types de zones temporelles. De cette, façon, il est possible d'obtenir une seule valeur de pente qui résume la période de non-contact et celle de contact entre le *Habitat* des deux *Espèces* en cause. Dans le cas de l'exemple, la première et la troisième zone sont des zones où il n'y a pas de contact entre les *Habitats*. Le Tableau 6 présente le calcul des moyennes des pentes.

Tableau 6 Moyenne des pentes pour les zones temporelles

Loup		Mouton	
Moyenne pour les zones temporelles de non-contact entre *Habitats*			
Nombre	$\frac{0+0}{2}=0$	Nombre	$\frac{0+0}{2}=0$
État de santé	$\frac{-1-1.5}{2}=-1.25$	État de santé	$\frac{1+1}{2}=1$

Dimension	$\frac{-2.5-1}{2}=-1.75$	Dimension	$\frac{13+4.5}{2}=8.75$

Les résultats de ces moyennes seront utilisés dans les calculs à venir lorsqu'il sera temps de décrire quantitativement les zones temporelles où il n'y a pas de contact entre les *Habitats*.

La dernière étape consiste à quantifier l'influence du contact entre les *Espèces* en utilisant les données du Tableau 6 et l'Équation 4. Le Tableau 7 présente les résultats de la quantification pour l'exemple.

Tableau 7 Quantification de la différence entre les pentes lors d'un contact et lors d'un non-contact

Loup		Mouton	
Moyenne pour les zones temporelles de non-contact entre *Habitats* (Q)			
Nombre	$\frac{0-0}{ABS(0)+ABS(0)}=0$	Nombre	$\frac{-0.05-0}{ABS(-0.05)+ABS(0)}=-1$
État de santé	$\frac{0.35+1.25}{ABS(0.35)+ABS(-1.25)}=1$	État de santé	$\frac{-0.3-1}{ABS(-0.3)+ABS(1)}=-1$
Dimension	$\frac{0.55+1.75}{ABS(0.55)+ABS(-1.75)}=1$	Dimension	$\frac{-1.25-8.75}{ABS(-1.25)+ABS(8.75)}=-1$

La quantification présentée au Tableau 7 montre la nocivité de l'interaction entre les deux *Espèces* pour chaque propriété calculée. Dans le cas du mouton, toutes les valeurs pointent vers un impact très négatif, car toutes les valeurs sont égales à -1 qui s'avère à être la plus grande valeur de nocivité. Pour le loup, l'interaction n'a pas d'influence sur le nombre d'*Organismes* vivants ($Q=0$). Par contre, le loup est influencé positivement par l'interaction en ce qui à-trait à l'état de santé et la dimension de l'*Habitat*.

Grace à cette méthode de quantification, il est donc possible d'observer l'influence d'une interaction sur la dimension des *Habitats*, le nombre d'individus et l'état de santé global d'une *Espèce*. Cette information sera utilisée dans le processus d'identification des interactions entre les espèces.

Algorithme de détection d'interactions

L'information fournie par les calculs préliminaires (section « Calculs préliminaires ») donne plusieurs indices sur les effets de l'interaction entre les paires d'*Espèces*. En effet, une quantification entre -1 et 1 est obtenue par l'analyse des variations des propriétés calculées soit : le nombre d'*Organisme*s vivants, l'état de santé global des *Espèces* en cause et la dimension de leur *Habitat*. Ces variations sont calculées en tenant compte de la valeur d'amplitude au début du contact entre les deux espèces ainsi que la valeur d'amplitude à la fin du contact. Malgré que cette information soit pertinente afin de comprendre l'impact qu'a chaque *Espèce* sur chaque autre *Espèce* en cours de simulation, une méthode englobant toutes les quantifications pour résumer l'interaction par un seul paramètre serait pertinent. C'est précisément ce que présente cette section. En effet, l'objectif ultime de l'algorithme de détection d'interactions consiste à identifier quel type d'interaction est en cause entre chaque paire d'*Espèces*. Pour ce faire, les valeurs quantifiées présentées à la section « Calculs préliminaires » sont fournies à un système à base de logique floue qui permettra d'identifier l'interaction en cause. Ce processus referme les deux étapes présentées à la Figure 48.

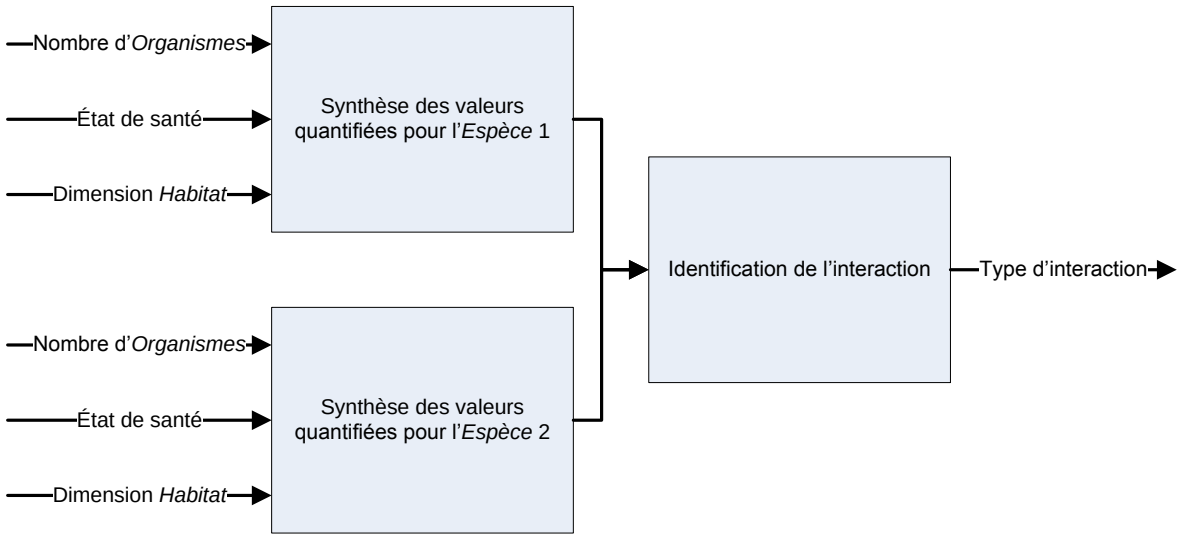


Figure 48 Système à base de logique floue pour l'identification du type d'interaction

La première étape consiste à prendre les résultats de la quantification obtenus à la section « Quantification des données temporelles » pour chacune des deux *Espèces* impliquées dans l'interaction et synthétiser cette information. La seconde étape prend le résultat de la première étape et l'analyse afin d'associer les valeurs

quantifiées à un type d'interaction qui sera fourni à l'utilisateur de la simulation visuelle interactive. Ces deux étapes sont détaillées dans les sections suivantes.

Synthèse des valeurs quantifiées

À ce stade, la quantification des trois propriétés calculées donne beaucoup d'information sur le comportement d'une *Espèce* lorsqu'elle est en contact avec une autre. En effet, leur analyse permet de connaître si le contact est bénéfique, neutre ou néfaste pour chaque propriété calculée. Par contre, tirer des conclusions sur l'analyse d'une seule propriété à la fois peut biaiser l'analyse. En effet, une seule valeur peut représenter la tendance concernant le type d'interaction, mais elle peut aussi proposer une implication erronée. Il est donc préférable de prendre toutes les propriétés calculées ensemble afin d'analyser quel type d'interaction est présente lors du contact entre les *Espèces*. Pour ce faire, les valeurs de quantification calculées pour chacune des deux *Espèces* sont synthétisées afin de connaître l'impact réel que le contact avec l'autre *Espèce* engendre. Les systèmes à base de logique floue présentés à gauche sur la Figure 48 (Synthèse des valeurs quantifiées pour l'Espèce 1 et Synthèse des valeurs quantifiées pour l'Espèce 2) permettent d'effectuer la synthèse des valeurs quantifiées.

Ces derniers utilisent trois entrées pour produire la sortie. Les trois entrées sont les valeurs quantifiées (résultats de l'algorithme de la section « Quantification des données temporelles ») pour l'*Espèce* à analyser soit le nombre d'*Organismes*, la dimension de l'*Habitat* et l'état de santé des *Organismes* de l'*Espèce* en question.

La fuzzification des entrées se fait en utilisant les ensembles flous présentés à la Figure 49.

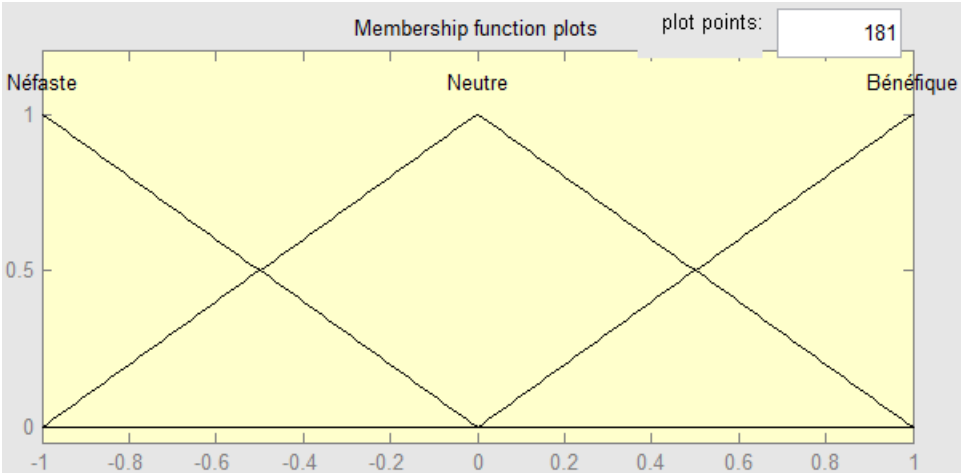
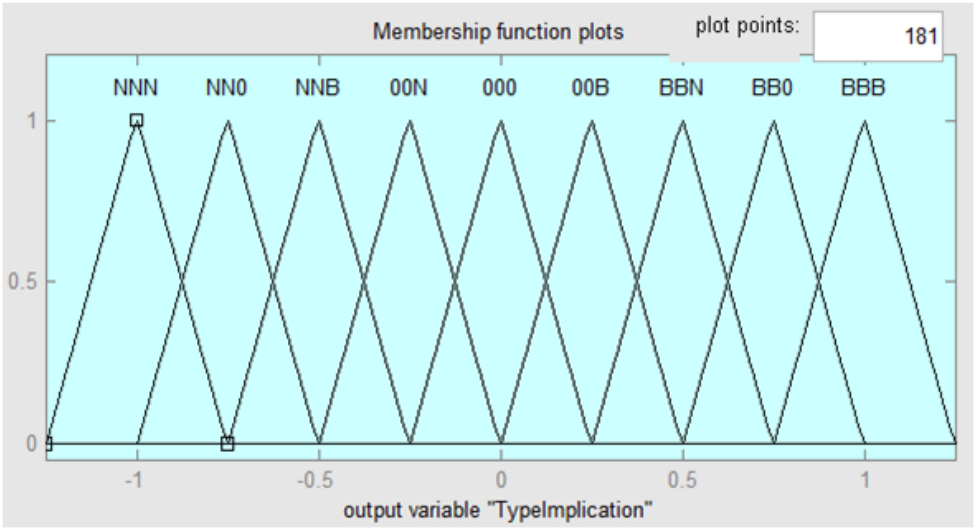
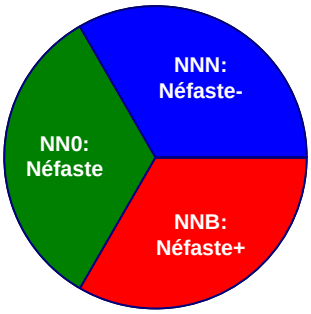


Figure 49 Ensembles flous pour les entrées du sytème de synthèse pour une *Espèce*

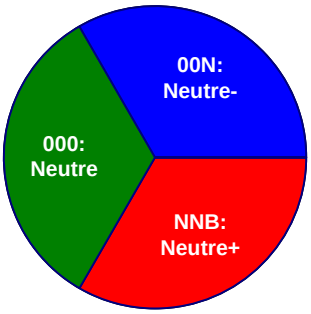
La Figure 49 montre qu'une entrée est néfaste si elle équivaut à -1, elle est neutre lorsqu'elle se situe à 0 et elle est bénéfique lorsqu'elle se situe à 1.

Les résultats possibles de la synthèse des trois propriétés calculées, donc de la sortie du système, sont représentés par les ensembles flous présentés à la Figure 50.

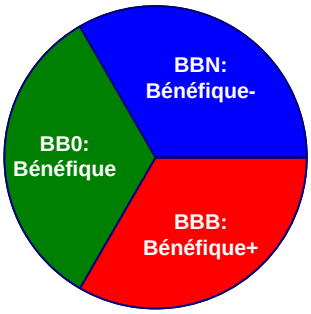




Division de l'ensemble flou Néfaste



Division de l'ensemble flou Neutre



Division de l'ensemble flou Bénéfique

Figure 50 Ensembles flous pour la sortie du système de synthèse pour une *Espèce*

À la base trois, ensembles flous peuvent être distingués. En effet, la synthèse de l'information fournie en entrée peut affirmer que l'implication du contact sur l'*Espèce* analysée est soit néfaste, neutre ou bénéfique. Ces trois ensembles sont ensuite fractionnés en trois sous-ensembles afin de permettre une plus grande granularité. Par exemple, l'ensemble flou *néfaste* devient *néfaste*+, *néfaste* et *néfaste*-. De cette façon, il est possible de classifier plus précisément le résultat de chaque règle. Les règles du système d'inférence sont présentées au Tableau 8.

Tableau 8 Règles d'inférence pour le système de synthèse pour une *Espèce*

NombreOrganismes	DimensionHabitat	ÉtatSantéOrgnanismes	TypeImplication
Néfaste	Néfaste	Néfaste	Néfaste-

Néfaste	Néfaste	Neutre	Néfaste
Néfaste	Néfaste	Bénéfique	Néfaste+
Néfaste	Neutre	Néfaste	Néfaste
Néfaste	Neutre	Neutre	Neutre-
Néfaste	Neutre	Bénéfique	Neutre
Néfaste	Bénéfique	Néfaste	Néfaste-
Néfaste	Bénéfique	Neutre	Neutre
Néfaste	Bénéfique	Bénéfique	Bénéfique-
Neutre	Néfaste	Néfaste	Néfaste
Neutre	Néfaste	Neutre	Neutre-
Neutre	Néfaste	Bénéfique	Neutre
Neutre	Neutre	Néfaste	Neutre-
Neutre	Neutre	Neutre	Neutre
Neutre	Neutre	Bénéfique	Neutre+
Neutre	Bénéfique	Néfaste	Neutre
Neutre	Bénéfique	Neutre	Neutre+
Neutre	Bénéfique	Bénéfique	Bénéfique
Bénéfique	Néfaste	Néfaste	Néfaste-
Bénéfique	Néfaste	Neutre	Neutre

Bénéfique	Néfaste	Bénéfique	Bénéfique-
Bénéfique	Neutre	Néfaste	Neutre
Bénéfique	Neutre	Neutre	Neutre+
Bénéfique	Neutre	Bénéfique	Bénéfique
Bénéfique	Bénéfique	Néfaste	Bénéfique-
Bénéfique	Bénéfique	Neutre	Bénéfique
Bénéfique	Bénéfique	Bénéfique	Bénéfique+

Ces règles permettent d'évaluer toutes les possibilités d'entrées pour le système. En effet, il y a deux entrées qui peuvent avoir trois valeur différentes, donc 27 possibilités de sorties pour le système. La Figure 51 présente un exemple de la façon dont les règles ont été créées.

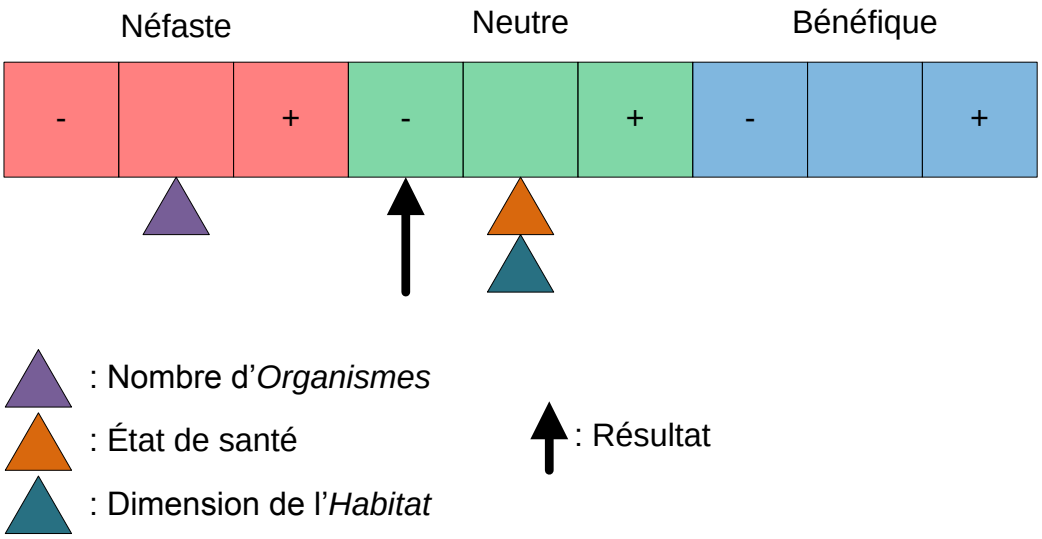


Figure 51 Exemple pour la création de règle d'inférence pour le système de synthèse d'une *Espèce*

Toutes les possibilités d'entrées ont été évaluées de la même façon afin de produire une sortie. En effet, les trois entrées sont tout d'abord associées à un qualificatif (néfaste, neutre ou bénéfique). Ensuite, elles sont

positionnées sur le graphique (les trois triangles sur la Figure 51). Finalement, le centre de masse est calculé afin de trouver qu'elle sera l'influence globale sur l'*Espèce* analysée (flèche noire sur la Figure 51). Dans l'exemple donné, la variation du nombre d'*Organismes* pour l'*Espèce* est néfaste, la variation de l'état de santé est neutre et la variation de la dimension de l'*Habitat* est aussi neutre. Donc, l'implication que l'interaction a sur l'Espèce est *neutre-*.

La synthèse des valeurs quantifiées permet de résumer l'impact sur les deux *Espèces* ayant eu un contact dans leur *Habitat* au cours de la simulation. En effet, cette synthèse permet de connaître l'influence des interactions sur les *Espèces*. Donc, il ne suffit à ce stade que d'identifier quel est le type d'interaction afin d'aviser l'utilisateur de la simulation visuelle interactive pour lui permettre d'augmenter sa compréhension du système simulé.

Identification de l'interaction

L'influence du contact entre *Habitats* sur les deux *Espèces* est désormais connue par l'analyse de la variation des propriétés calculées (section « Propriétés calculées ») et la synthèse des propriétés calculées pour chacune des *Espèces* (section « Synthèse des valeurs quantifiées »). De cette façon, l'utilisateur de la simulation visuelle interactive peut désormais connaître l'impact que les *Espèces* faisant partie de l'*Écosystème* simulé ont les unes sur les autres. Par contre, il est possible de simplifier l'information à transmettre à l'utilisateur afin de résumer chaque interaction par un seul qualitatif. Ce dernier est en fait le nom d'une interaction biologique qui englobe l'information nécessaire pour comprendre l'implication des interactions (Tableau 1). Cette section présentera le processus basé sur un système à base de logique floue qui prend en entrée les valeurs de synthèse (section « Synthèse des valeurs quantifiées ») pour une paire d'*Espèces* et retourne le nom de l'interaction identifiée par le Framework. Le système à base de logique floue est présenté à droite sur la Figure 48 (Identification de l'interaction).

Le système comporte deux entrées. Chacune d'entre elles représente la synthèse des données quantifiées pour les deux Espèces. Les valeurs que peuvent prendre les entrées sont présentées à la Figure 52.

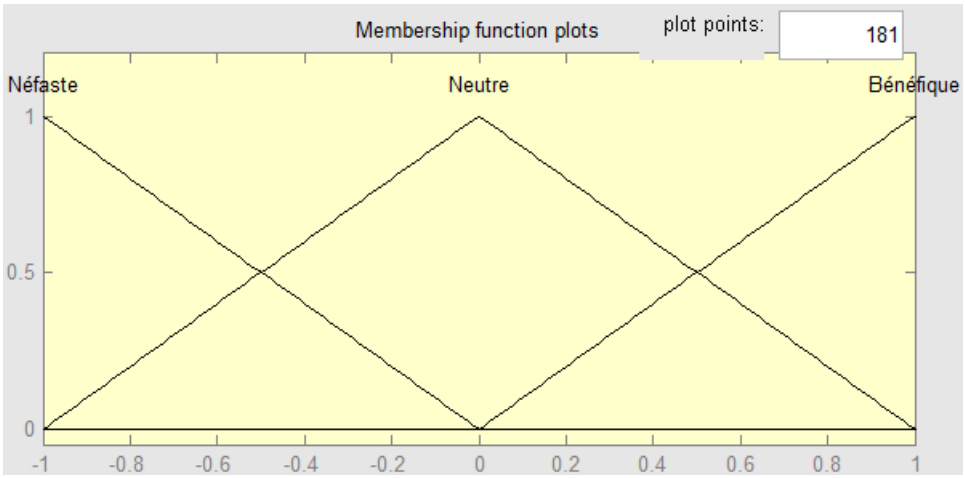


Figure 52 Valeurs possibles pour une entrée du système à base de logique floue permettant d'identifier le type d'interaction entre deux *Espèces*

L'entrée peut faire partie de trois ensembles flous soit *néfaste*, *neutre* ou bénéfique. Les entrées sont fournies par le système de synthèse présenté à la section « Synthèse des valeurs quantifiées ». Il est à noter qu'il n'est pas nécessaire de diviser les ensembles flous pour avoir la même granularité que la sortie du système de synthèse. En effet, pour l'identification de l'interaction, fractionner les ensembles flous pour être plus précis n'est pas nécessaire, car le système a seulement besoin de savoir à quel des trois ensembles flous l'entrée appartient. C'est pour cette raison que le système d'identification ne comporte que trois ensembles flous pour représenter les entrées. Les valeurs possibles pour la sortie du système d'identification d'interaction sont présentées à la Figure 53.

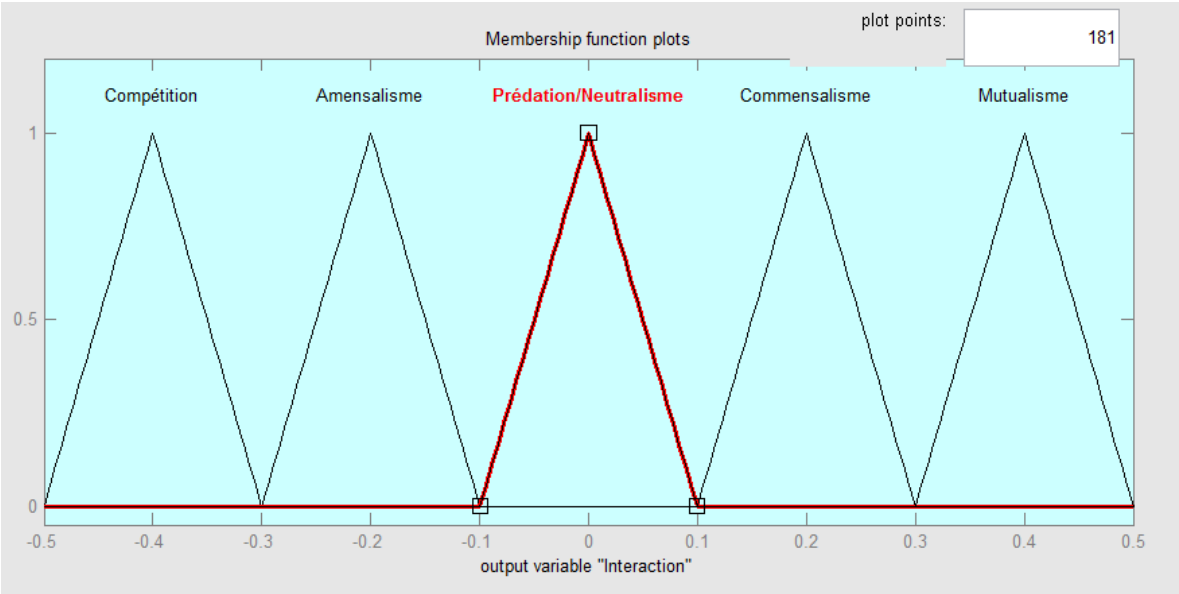


Figure 53 Valeurs pour la sortie du système à base de logique floue permettant d'identifier le type d'interaction entre deux Espèces

Les intervalles de valeurs de sorties sont au nombre de six et représentent chacune une interaction. La valeur de sortie va de l'interaction le plus néfaste, c'est-à-dire la *Compétition* qui est considérée ainsi, car les deux *Espèces* en cause ont des répercussions négatives dues à la présence de l'autre *Espèce* à l'interaction la plus bénéfique soit le *Mutualisme*. Il est à noter que les interactions *Neutralisme* et *Prédation* occupent le même intervalle. La raison de ce chevauchement est présentée par la Figure 54 qui montre les transitions entre les interactions pour le système d'identification.

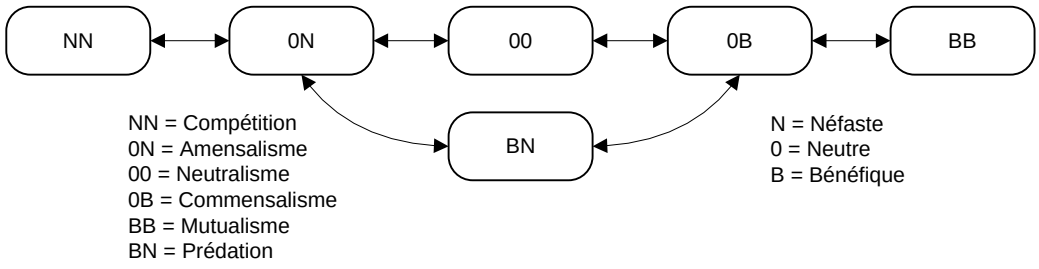


Figure 54 Transitions entre les différents types d'interaction

Chaque état sur la Figure 54 représente une interaction qui est identifiée par le degré de nocivité (B pour bénéfique, 0 pour neutre et N pour néfaste) pour chaque Espèce. Il est à noter que les deux degrés de

nocivité peuvent être permutés, par exemple B0 équivaut à 0B. Pour qu'un état change, un seul degré de nocivité peut changer. En prenant de ce principe, le *Neutralisme* et la *Prédation* se retrouvent au même endroit soit au centre. Pour les différencier, un autre système à base de logique floue sera présenté sous peu. Les règles pour détecter le type d'interaction impliqué entre deux Espèces sont données au Tableau 9.

Tableau 9 Règles d'inférence pour le système d'identification du type d'interaction entre deux *Espèces*

Espèce1	Espèce2	Interaction
Néfaste	Néfaste	Compétition
Néfaste	Neutre	Amensalisme
Néfaste	Bénéfique	Prédation
Neutre	Néfaste	Amensalisme
Neutre	Neutre	Neutralisme
Neutre	Bénéfique	Commensalisme
Bénéfique	Néfaste	Prédation
Bénéfique	Neutre	Commensalisme
Bénéfique	Bénéfique	Mutualisme

C'est règles sont dérivées directement de la définition des interactions qui a été donnée à la section « Modèle conceptuel ».

Le système à base de logique floue servant à identifier le type d'interaction ne permet pas de différencier le *Neutralisme* de la *Prédation*. En effet, la *Prédation* et le *Neutralisme* occupent le même intervalle pour ce système (Figure 53). Donc, un autre système à base de logique floue a été créé pour informer l'utilisateur lorsque l'interaction est de type prédation. Ce dernier est présenté à la Figure 55.

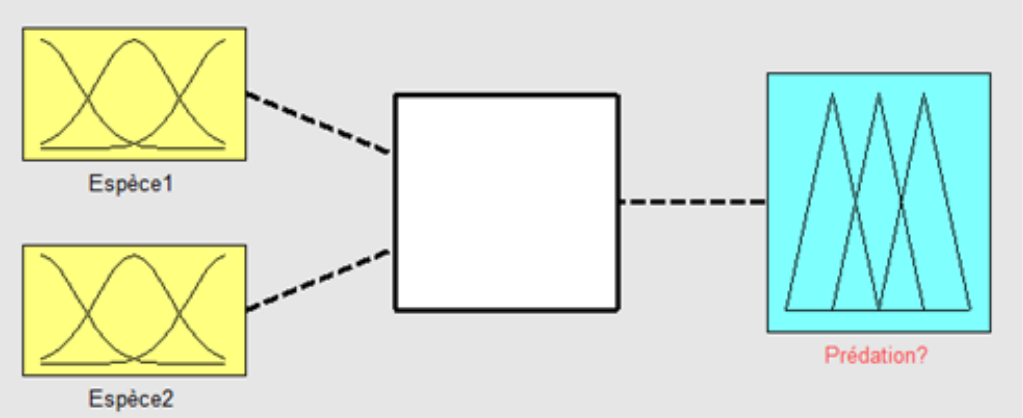


Figure 55 Système à base de logique floue pour détecter lorsqu'il y a *Prédation* entre deux *Espèces*

Les entrées sont les valeurs de synthèse pour les deux *Espèces* qui ont été obtenues à l'aide du système à base de logique floue présenté à la section « Synthèse des valeurs quantifiées ». Elles se divisent en trois ensembles flous présentés à la Figure 56.

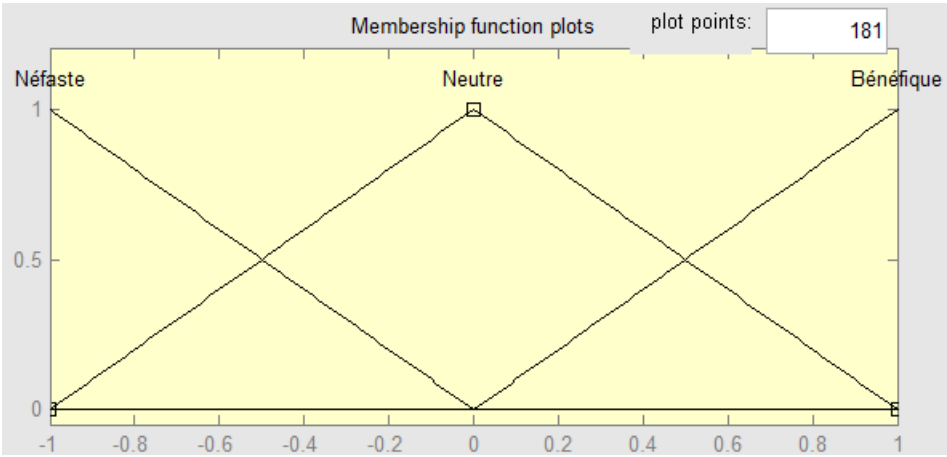


Figure 56 Ensembles flous pour les entrées d'un système à base de logique floue permettant d'identifier lorsqu'il y a *Prédation*

Pour sa part, la sortie de ce système peut prendre deux valeurs soit Oui ou Non. Les valeurs de la sortie sont présentées à la Figure 57.

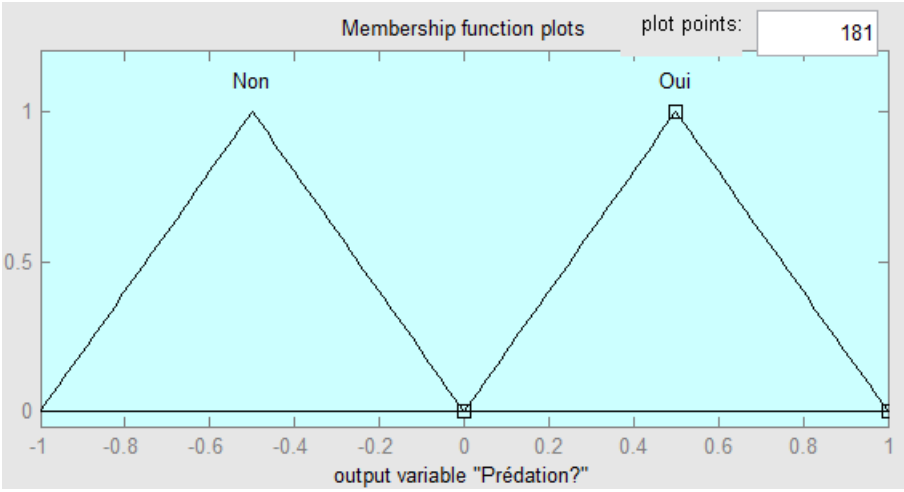


Figure 57 Valeur de sortie pour un système à base de logique floue permettant didentifier lorsqu'il y a *Prédation*

Le calcul de la sortie du système en fonction des deux entrées est dicté par les règles d'inférence présentées au Tableau 10.

Tableau 10 Règles d'inférence pour un système à base de logique floue permettant didentifier lorsqu'il y a *Prédation*

Espèce1	Espèce2	Prédation?
Néfaste	Bénéfique	Oui
Bénéfique	Néfaste	Oui
Autre	Autre	Non

Les deux derniers systèmes à base de logique floue permettent d'identifier quel type d'interaction est présente entre chaque *Espèce* d'un *Écosystème*. Donc, l'objectif d'informer l'utilisateur du type d'interaction en jeu lors du déroulement d'une simulation visuelle interactive est atteint. Le prochain chapitre présentera différents scénarios qui démontreront le fonctionnement du Framework dans son ensemble.

Chapitre 4 : Scénarios, résultats et discussion

Le « Chapitre 2 : Modèle conceptuel et architecture » a présenté un modèle conceptuel basé sur le paradigme des écosystèmes ainsi que l'architecture logicielle qui y est associée. Le « Chapitre 3 : Détection d'interactions » a présenté la détection d'interactions entre entités en cours de simulation. Le présent chapitre montre le Framework en application. En effet, cette section présentera tout ce qui a à trait aux scénarios utilisés pour valider le Framework et la détection d'interactions. Tout d'abord, l'engin de simulation NetLogo [89] est présenté. En effet, NetLogo est utilisé pour générer les résultats de simulation multi-agents qui seront ensuite acheminés au Framework. Ensuite, les trois scénarios qui permettront de valider la fonctionnalité du Framework sont présentés. Finalement, les résultats fournis par les scénarios de simulation seront analysés et discutés.

Engin de simulation multi-agents NetLogo

Le Framework proposé est utilisé afin de simplifier la tâche d'analyse de l'utilisateur d'une simulation visuelle interactive de système complexe. Le Framework ne génère donc pas de données brutes, mais il restructure, les analyses et donne des pistes de compréhension à l'utilisateur. C'est donc pour cette raison qu'un engin de simulation est nécessaire pour générer les résultats qui sont analysés par le Framework proposé. Cette section présentera brièvement le positionnement de l'engin de simulation NetLogo utilisé dans le cadre des travaux de thèse.

La Figure 58 montre le flux des données de simulation générées par l'outil de simulation NetLogo.

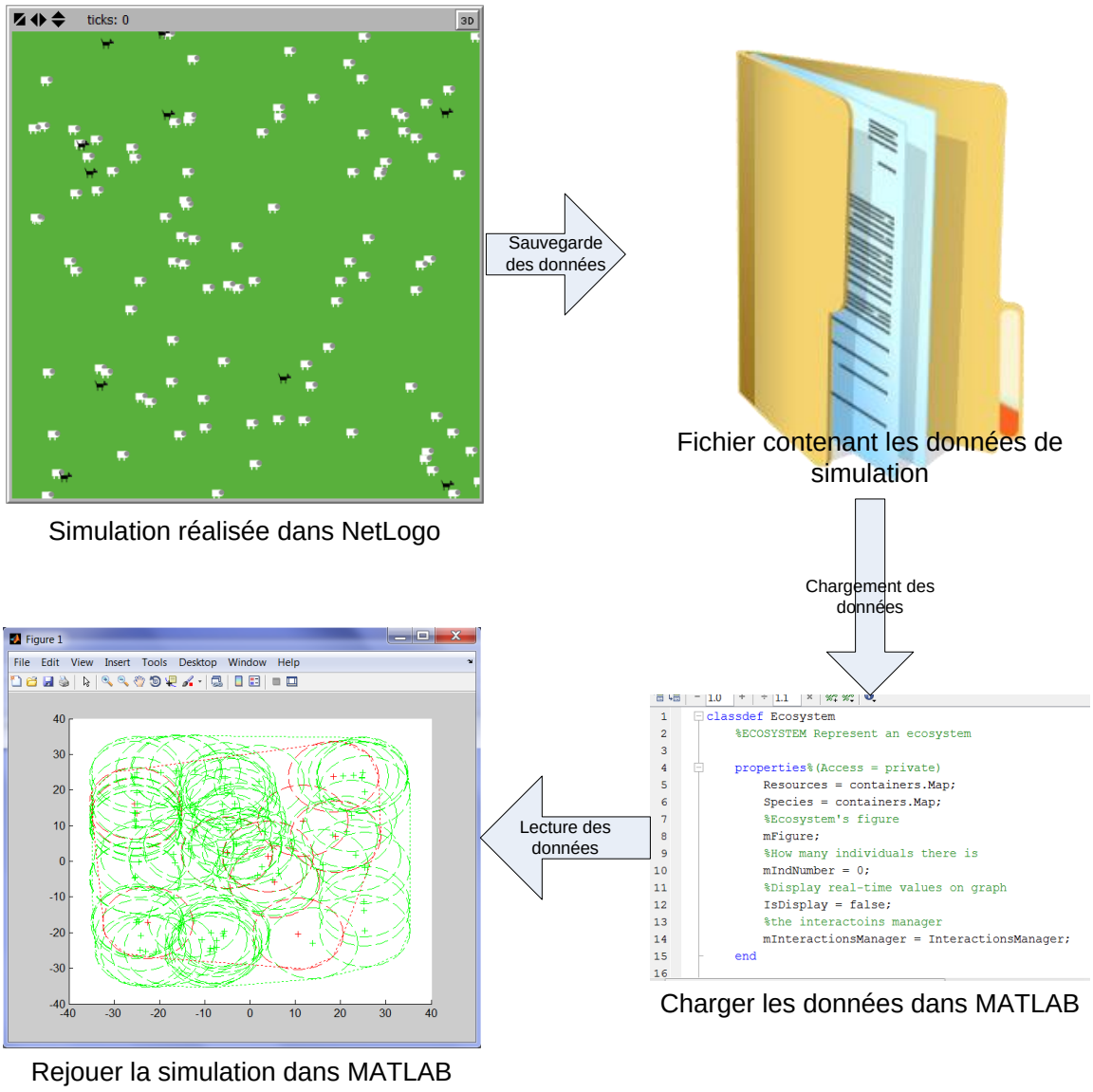


Figure 58 Le flux de données généré par l'outil de simulation multi-agents NetLogo

Comme la Figure 58 le démontre, les simulations utilisées sont exécutés en deux temps. Dans un premier temps, la simulation est exécutée sans rétroaction de la part du Framework. Ce qui implique qu'à ce stade, le Framework n'est pas encore impliqué dans la boucle de simulation. Dans un second temps, les données de simulation sont importées dans Matlab afin d'être en mesure d'utiliser les fonctionnalités du Framework. Donc, à ce moment les données de simulation obtenues par NetLogo sont changées dans le Framework. C'est à ce moment que les algorithmes de détection et d'identification d'interactions sont sollicités. De cette façon, il a été

possible de découpler au maximum les différents composants utilisés. De plus, les efforts d'intégration ont pu être éliminés car le but de la thèse est de montrer une preuve de concept et non un logiciel intégré.
Les scénarios présentés dans les prochaines sections sont tous exécutés en respectant le flux des données présenté à la Figure 58.

Scénarios

Le Framework peut être divisé en deux parties qui doivent être mises à l'essai. D'une part, l'architecture basée sur le modèle conceptuel présenté au « Chapitre 2 : Modèle conceptuel et architecture » doit être validée. D'autre part, les résultats fournis par l'algorithme de détection d'interactions présenté à la section « Algorithme de détection d'interactions » doivent être analysés. Pour ce faire, plusieurs scénarios ont été développés. Cette section présente donc ces scénarios qui ont été réalisés avec l'objectif principal de valider l'approche proposée. Chacun d'entre eux a été réalisé afin d'évaluer une fonctionnalité précise du Framework. Le premier est un scénario de prédation entre des loups et des moutons. Il permet d'analyser le comportement du Framework pour une même simulation lancée à plusieurs reprises. Le second scénario présente l'évolution d'une invasion de zombies. Ce dernier permet d'évaluer les résultats générés par l'algorithme de détection d'interaction lors de la simulation d'un scénario complexe. Le dernier scénario présente l'évolution des plusieurs bactéries dans un même environnement. Il permet d'évaluer le comportement d'ensemble du Framework lorsqu'un utilisateur peut interagir avec la simulation en cours.

Scénario de prédation

Le scénario de prédation a été réalisé pour deux raisons. D'une part, il permet d'évaluer l'algorithme de détection d'interactions. En effet, sa simplicité introduira un minimum d'imprévisibilité dans les résultats anticipés. D'autre part, il sera utilisé afin d'évaluer la reproductibilité des résultats. Une fois de plus, sa simplicité et prévisibilité vont permettre d'effectuer plusieurs simulations de ce même scénario et de comparer les résultats entre eux. Donc, il sera possible d'observer facilement si l'algorithme fournit des résultats cohérents qui sont au diapason avec l'analyse d'un expert du domaine simulé.

Le scénario de prédation permet d'observer l'interaction entre les loups et les moutons lorsque ces deux *Espèces* sont insérées dans un même écosystème. La structure statique de l'*Écosystème* est présentée à la Figure 59. Les Figure 60 et Figure 61 présentent, pour leur part, l'organigramme des *Organismes* de chaque *Espèce*.

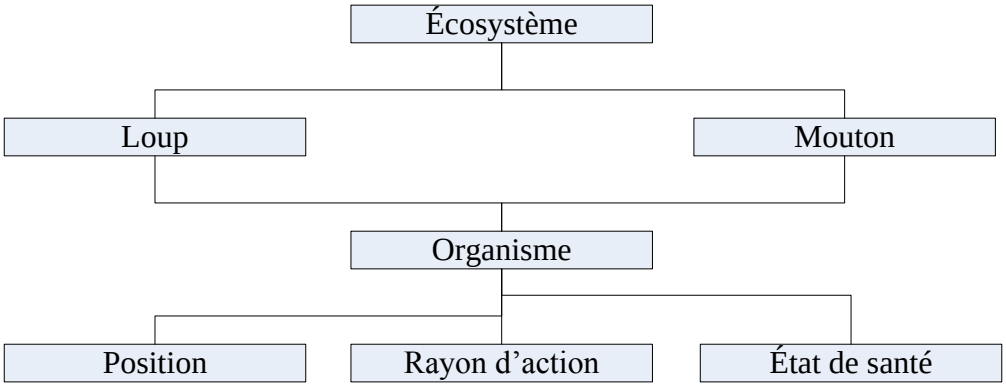


Figure 59 Structure statique du scénario de prédation

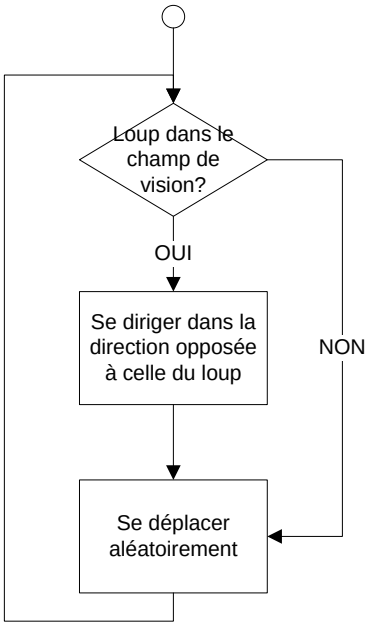


Figure 60 Organigramme de comportement d'un mouton

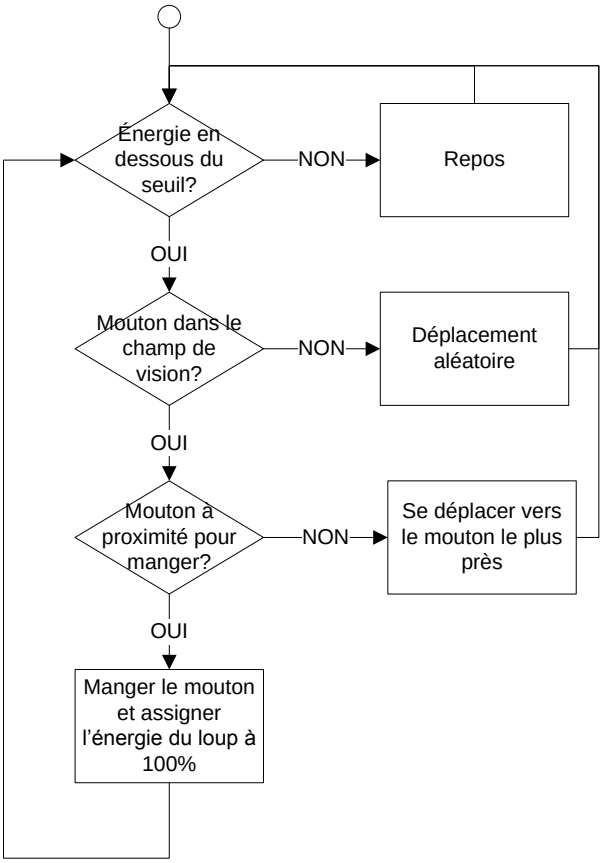


Figure 61 Organigramme d'un loup

Dans ce scénario, les états pour le mouton sont les suivants (Figure 60) : se sauver des loups ou se déplacer aléatoirement. Donc, le mouton se déplace aléatoirement sauf au moment où un loup entre dans son champ de vision. Dans ce cas, le mouton se dirige à l'opposé du loup détecté. Il est à noter que les moutons n'ont pas à se chercher de la nourriture. Donc, la seule façon qu'ils peuvent périr est lorsqu'ils se font dévorer par un loup.

Dans le cas du loup (Figure 61), il doit se nourrir pour survivre. C'est donc pour cette raison qu'il doit chasser les moutons. L'organigramme de la Figure 61 présente les états décrivant le comportement pour le loup. Tout d'abord, le loup reste au repos et conserve son énergie s'il n'est pas nécessaire pour lui de s'alimenter. Il est à noter que pour un loup, être au repos consomme moins d'énergie que lorsqu'il est en mouvement. Donc, le niveau d'énergie du loup doit descendre sous un seuil, choisi par l'utilisateur en début de simulation, afin de débuter sa chasse. Lorsque ce seuil est franchi, le loup part à la recherche de proie (i.i. des moutons) pour s'en nourrir. Il va de soi que le loup meurt après un certain temps s'il ne trouve aucune source de nourriture.

Lorsque le loup chasse, il regarde tout d'abord dans son champ de vision s'il n'y a pas la présence d'une proie. Si c'est le cas, il se déplace dans la direction du mouton le plus près de lui. Si ce n'est pas le cas, il recherche des proies en se déplaçant aléatoirement. Finalement, lorsqu'un mouton est à la portée d'un loup, ce dernier le dévore et son énergie est remise à sa capacité maximale. Il est à noter que le champ de vision du mouton est plus grand mais moins profond que celui du loup. De plus, la vitesse du loup est plus grande que celle du mouton. Ces éléments sont paramétrables par l'utilisateur en début de simulation.

Scénario d'invasion de zombies

Le scénario d'invasion de zombies a comme principal objectif de démontrer le comportement de l'algorithme de détection d'interaction dans le cas de plus de deux *Espèces*. En effet, ceci permettra d'évaluer si les interactions entre deux *Espèces* sont cohérentes malgré la présence d'une troisième *Espèce*. Donc, le scénario évaluera si l'assignation du type d'interaction est valide dans un cas complexe.

Le scénario permet de suivre l'évolution d'une invasion de zombies. La structure statique de ce scénario est présentée à la Figure 62. Ce dernier renferme trois *Espèces* dont le comportement est présenté aux Figure 63, Figure 64 et Figure 65.

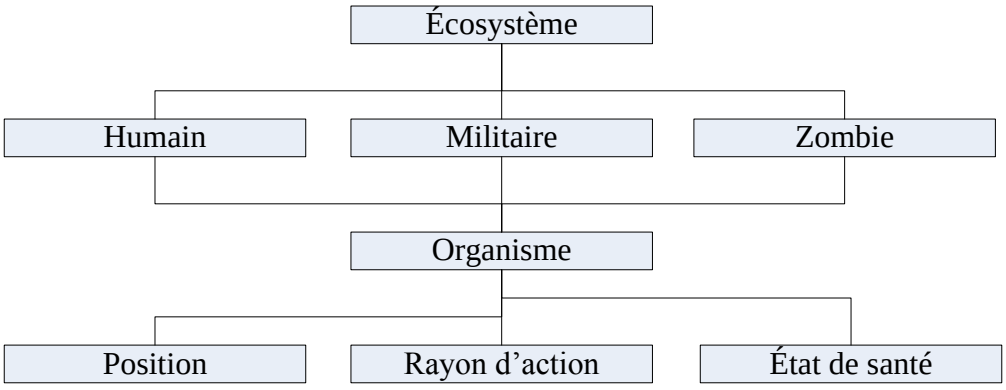


Figure 62 Structure statique pour le scénario d'invasion de zombies

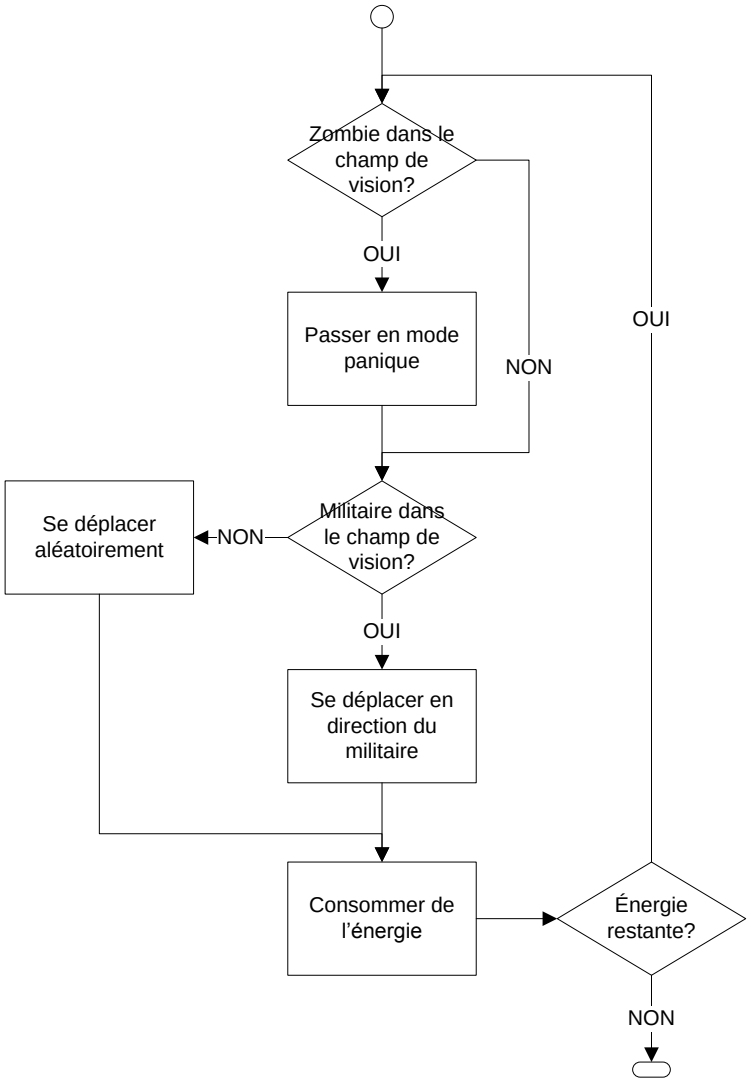


Figure 63 Organigramme du comportement d'un humain dans le scénario d'invasion de zombies

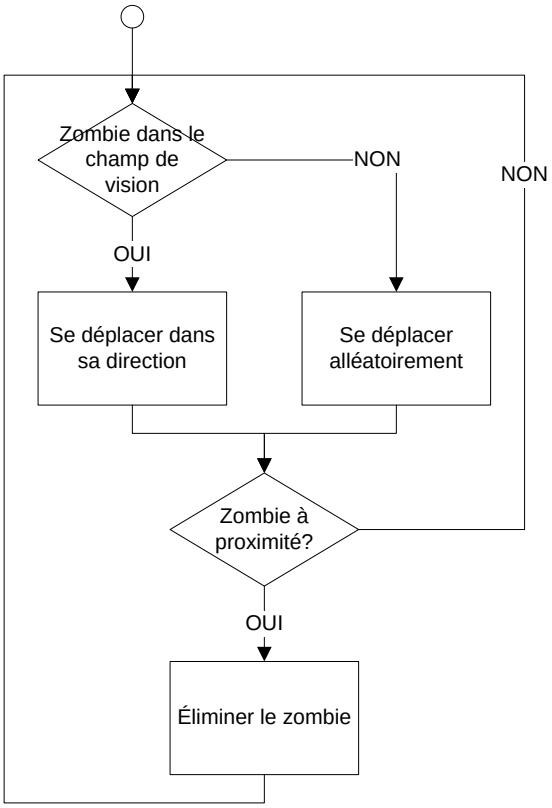


Figure 64 Organigramme du comportement d'un militaire dans le scénario d'invasion de zombie

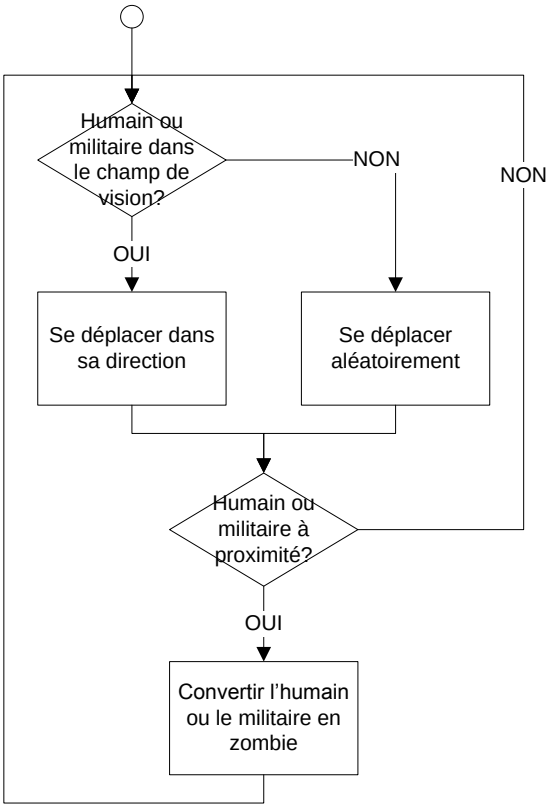


Figure 65 Organigramme du comportement d'un zombie dans le scénario d'invasion de zombie

Les trois *Espèces* du scénario sont les humains, les militaires et les zombies. Les humains (Figure 63) on comme objectif principal de survivre à l'invasion. Pour ce faire, ils peuvent adopter trois comportements. Le premier consiste à se sauver des zombies, car c'est à leur contact que les humains deviennent infectés. Le second comportement consiste à se diriger vers le militaire le plus près, car ces derniers sont armés et peuvent protéger les humains des zombies. Le dernier agissement possible pour un humain se produit lorsqu'il voit un zombie. À ce moment, l'humain passe en mode panique et fuit le zombie observé. Il est à noter que lorsqu'un humain est en panique, il consomme plus d'énergie et que ceci peut mener à la mort par épuisement.

Les militaires (Figure 64) ont comme objectif principal d'éliminer tous les zombies. Pour ce faire, ils peuvent adopter trois états. Le premier consiste à la recherche de zombies. Dans le cas où il n'y a pas de zombie dans le champ de vision d'un militaire, ce dernier se déplace de façon aléatoire. Le deuxième état est la poursuite. En effet, dans cet état, le militaire tente de rattraper le zombie le plus près qui se trouve dans son champ de vue. Le dernier état consiste en l'élimination du zombie. Cette étape comporte des risques pour le militaire, car

c'est parfois (pourcentage de chance de succès préétablie par l'utilisateur) le zombie qui l'emporte et infecte le militaire. Dans le cas contraire, l'effort requis par le militaire pour éliminer le zombie engendre une perte d'énergie pour le militaire. Donc, tout comme l'humain, le militaire peut mourir d'épuisement s'il y a trop de zombies à éliminer.

Les zombies (Figure 65) ont comme unique objectif d'infecter tous les autres *Organismes*. Pour ce faire, ils peuvent agir de trois façons. La première est lorsqu'un zombie ne voit aucun autre *Organisme*. À ce moment, il passe en mode chasse afin de trouver des proies en se déplaçant aléatoirement. Lorsqu'un autre *Organisme* est aperçu, le zombie se dirige en sa direction. Finalement, si le zombie attrape un *Organisme* non infecté, il lui transmet le virus qui le transformera en zombie.

Scénario d'évolution de bactéries

Le scénario d'évolution de bactéries à comme principal objectif d'évaluer le comportement du Framework dans le cas où il est possible à un utilisateur d'interagir avec la simulation en cours. En effet, ce scénario a été développé de façon à ce qu'un utilisateur puisse ajouter des *Organismes* d'une nouvelle *Espèce* en cours de simulation. De cette façon, il sera possible d'observer le comportement de l'algorithme de détection d'interaction ainsi que la stabilité du Framework dans un cas de simulation visuelle interactive d'un système complexe.

Le scénario de bactéries s'inspire de l'article de Nahum et al. [90]. Il consiste à analyser le développement d'une communauté bactérienne intransitive[5] de type *Roche-Papier-Ciseau*. Une seule variante a été apportée au scénario de Nahum. Cette dernière vise à inclure l'utilisateur dans la boucle de simulation. En effet, c'est ce dernier qui décide du moment où introduire la bactérie *Ciseau* à l'*Écosystème*. De plus, l'introduction de cette bactérie change le comportement de la bactérie *Papier* qui deviendra la bactérie *Papier'*. Ce changement de comportement en cours de simulation permettra d'observer l'adaptabilité du Framework. Donc, ceci permettra d'évaluer le changement d'interaction entre la *Roche* et le *Papier* dû à l'introduction du *Ciseau*. La structure statique du scénario est présentée à la Figure 66 et les organigrammes pour chaque bactérie sont présentés à la Figure 67, la Figure 68 et la Figure 69.

[5] Se dit d'une relation R telle que : si (x R y) et si (y R z), alors (x R z) est obligatoirement *faux*. [93]

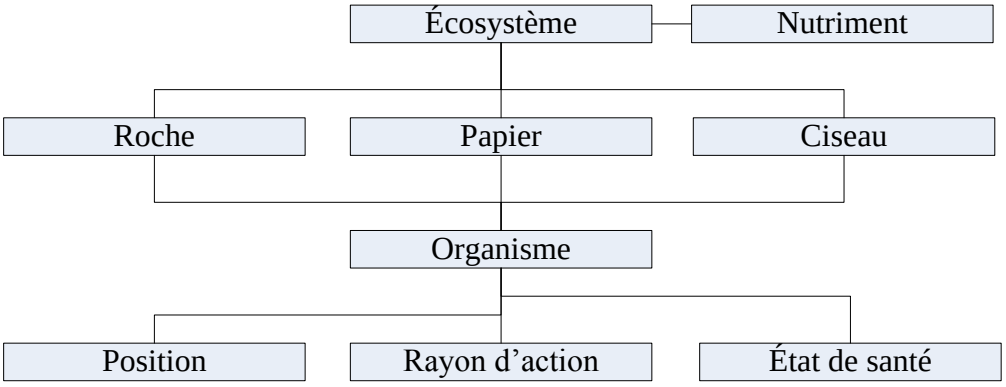


Figure 66 Structure statique du scénario d'évolution de bactéries

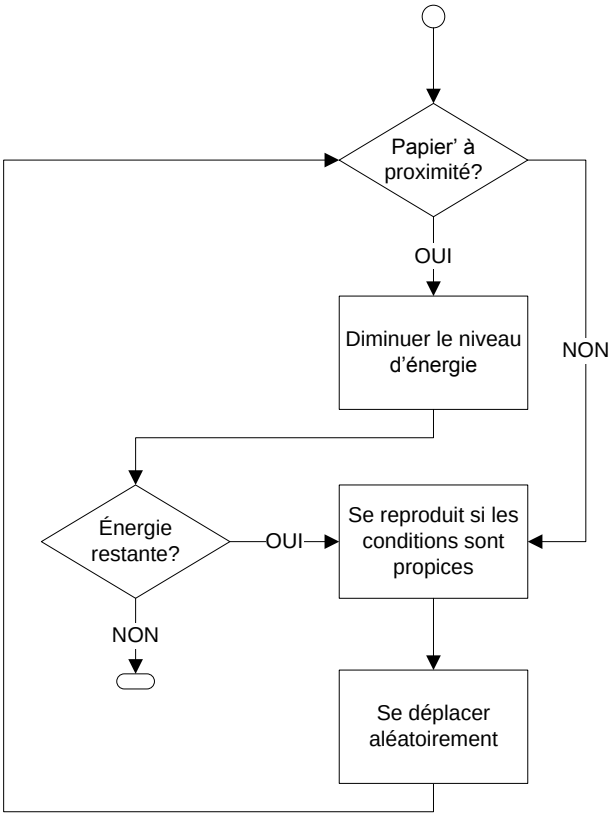


Figure 67 Organigramme du comportement d'une *Roche*

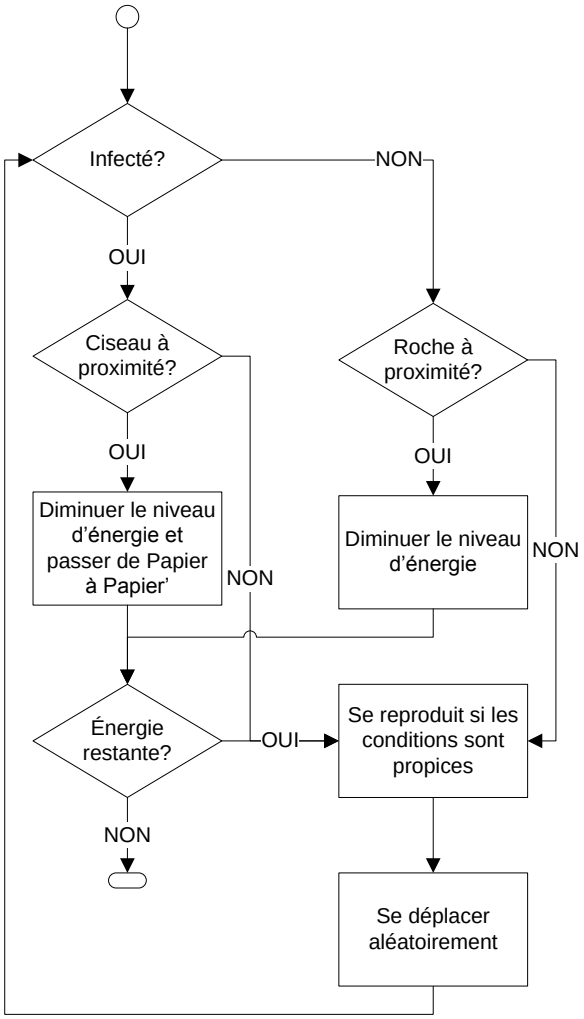


Figure 68 Organigramme de comportement du *Papier*

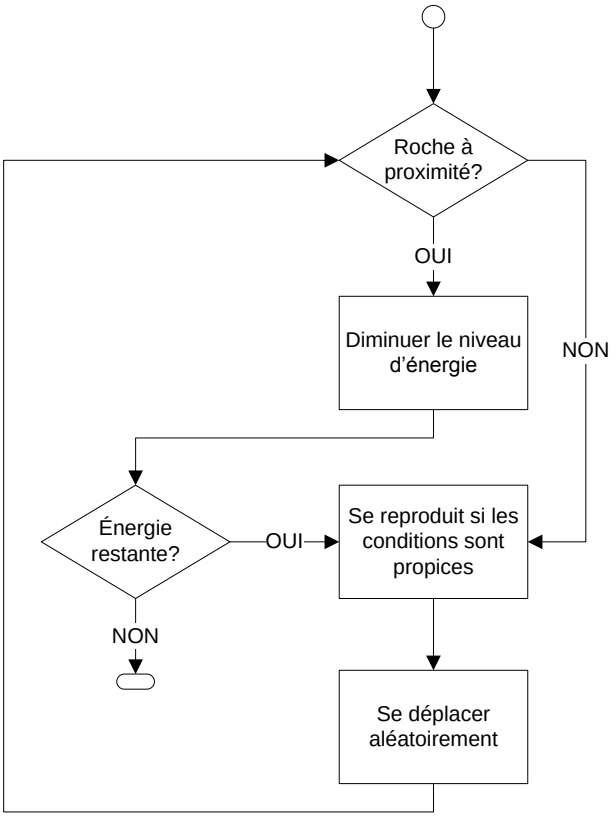


Figure 69 Organigramme de comportement pour le *Ciseau*

La Figure 67 présente le comportement pour la bactérie *Roche*. Son comportement se divise en trois parties. Tout d'abord, elle réagit à la présence de la bactérie *Papier'* si elle se trouve à proximité. Dans ce cas, *Papier'* affecte la *Roche* en l'affaiblissant. Ensuite, si la bactérie possède suffisamment d'énergie, elle exécute une division cellulaire [91] afin de se reproduire. Cette opération a un coût énergétique qui est préalablement défini par l'utilisateur. Dans le cas contraire, elle va cesser d'exister. Il est à noter que des nutriments sont répartis uniformément sur le sol de l'*Écosystème* et ils permettent aux bactéries de refaire le plein d'énergie lorsque c'est nécessaire. Finalement, la bactérie se déplace de façon aléatoire dans l'*Écosystème*.

Le comportement de la bactérie *Papier* est présenté à la Figure 68. Dû au fait que son comportement change en présence de la bactérie *Ciseau*, l'organigramme se divise en deux parties. D'une part, avant l'arrivée de la bactérie *Ciseau*, le *Papier* réagit négativement à la présence de *Roche* par la diminution de son taux d'énergie. D'autre part, lors de l'insertion de la bactérie *Ciseau* dans l'*Écosystème*, la bactérie *Papier* devient *Papier'*. Dans ce cas, elle est désormais néfaste à la bactérie *Roche* tout en devenant immunisée à cette

dernière. Dans les deux cas, l'organigramme se termine par la reproduction de la bactérie et par un déplacement aléatoire.

L'organigramme pour la bactérie *Ciseau* est présenté à la Figure 69. Le *Ciseau* est affecté négativement par la *Roche*. Donc l'organigramme se résume en 3 étapes. La première consiste à vérifier s'il y a présence d'une *Roche* à proximité. Si c'est le cas, l'énergie du *Ciseau* s'en voit diminuée. La seconde étape consiste à l'étape de la reproduction. Si la bactérie a suffisamment d'énergie pour se reproduire, elle va le faire. Finalement, le *Ciseau* se déplace de façon aléatoire lui aussi.

Les trois scénarios élaborés vont permettre d'évaluer le Framework de trois façons. Le premier va permettre de vérifier le fonctionnement de la détection d'interactions. De plus, il va permettre de vérifier que la simulation répétitive d'un même scénario donne des résultats de détection d'interactions identiques. Le deuxième évaluera le résultat de la détection d'interactions lors qu'il y a plus de deux *Espèces*. Finalement, le dernier scénario va permettre d'évaluer la détection d'interactions lorsqu'un utilisateur intervient dans la boucle de simulation. Les résultats des simulations des trois scénarios sont présentés à la section suivante.

Analyse des résultats des scénarios de simulation

Cette section présente les résultats de simulation pour les scénarios présentés à la section « Scénarios ». De plus, une discussion suivra chaque résultat afin d'expliquer la valeur de sortie obtenue pour chaque scénario. Cette section tentera donc de présenter la validité de l'approche proposée en montrant la cohérence des résultats obtenus en fonction du comportement des *Organismes*. Tout d'abord, le scénario de prédation sera utilisé pour montrer la validité de l'approche pour un scénario simple composé de seulement deux *Espèces*. De plus, il montrera que les résultats sont reproductibles pour une même simulation exécutée à plusieurs reprises. Ensuite, l'invasion de zombies permettra de présenter le résultat de la détection d'interactions pour un cas complexe. Finalement, le scénario de bactéries décrira les résultats d'une simulation avec une intervention de la part d'un utilisateur. Il est à noter que la résolution temporelle des graphiques et des tableaux présentés dans cette section est au 50 pas de temps. En effet, ceci permet d'alléger les graphiques et les tableaux. De plus, l'algorithme de détection d'interactions est seulement lancé à la fin d'un contact entre les *Habitats* des *Espèces*. Donc, il n'est pas nécessaire d'avoir une résolution à chaque pas de temps, car un contact entre *Espèces* peut durer plusieurs pas de temps.

Scénario de prédation

Cette section analysera les résultats du scénario de prédation en deux parties entre les loups et les moutons présenté à la section « Scénario de prédation ». D'une part, une simulation sera présentée afin d'évaluer l'algorithme de détection d'interactions. D'autre part, une présentation de la convergence de la valeur de l'interaction pour le scénario des moutons et de loups sera présentée par le biais des résultats de plusieurs simulations.

Les paramètres initiaux pour la génération des résultats sont présentés dans le Tableau 11 :

Tableau 11 Paramètres initiaux pour les simulations du scénario de prédation

Écosystème	
Dimension (unité[6])	70x70
Loup	
Nombre	2
Profondeur du champ de vue (unités)	7
Angle de vision	180˚
Énergie consommée au repos	0.25 de l'énergie restante
Énergie consommée pour se déplacer	0.75 de l'énergie restante
Vitesse (unités/pas de temps)	5
Mouton	
Nombre	10

[6] Une unité est la mesure de longueur pour un scénario. Il n'y a pas de conversion directe entre « unité » et « mètre ».

Profondeur du champ de vue (unités)	5
Angle de vision	300˚
Vitesse (unités/pas de temps)	3

Lors de l'exécution d'un tel scénario, l'utilisateur s'attend à observer un phénomène de *Prédation* où les loups dévorent les moutons jusqu'à leur extinction. En effet, c'est le comportement que l'on retrouve dans la nature lorsque ces deux *Espèces* sont confrontées. L'exemple qui sera présenté dans ce qui suit montre une simulation représentative pour un tel type de scénario. La Figure 70 présente la valeur de l'interaction détectée par le Framework pour un l'exécution du scénario de prédation entre des moutons et des loups. La figure présente la valeur du résultat de la synthèse[7] (section « Synthèse des valeurs quantifiées ») pour les moutons en bleu et pour la synthèse des loups en rouge. De plus la valeur de l'interaction est présentée en mauve. Ces mêmes données peuvent être retrouvées dans le Tableau 12. Le Tableau 13 présente pour sa part le moment précis du décès des *Organismes* au cours de la simulation. Un extrait des données brutes est montré au Tableau 14.

[7] Le résultat de la synthèse permet de regrouper les propriétés calculées (le nombre d'*Organisme*s vivants, l'état de santé global de l'*Espèces* et la dimension de l'*Habitat*.) en une seule valeur quantitative.

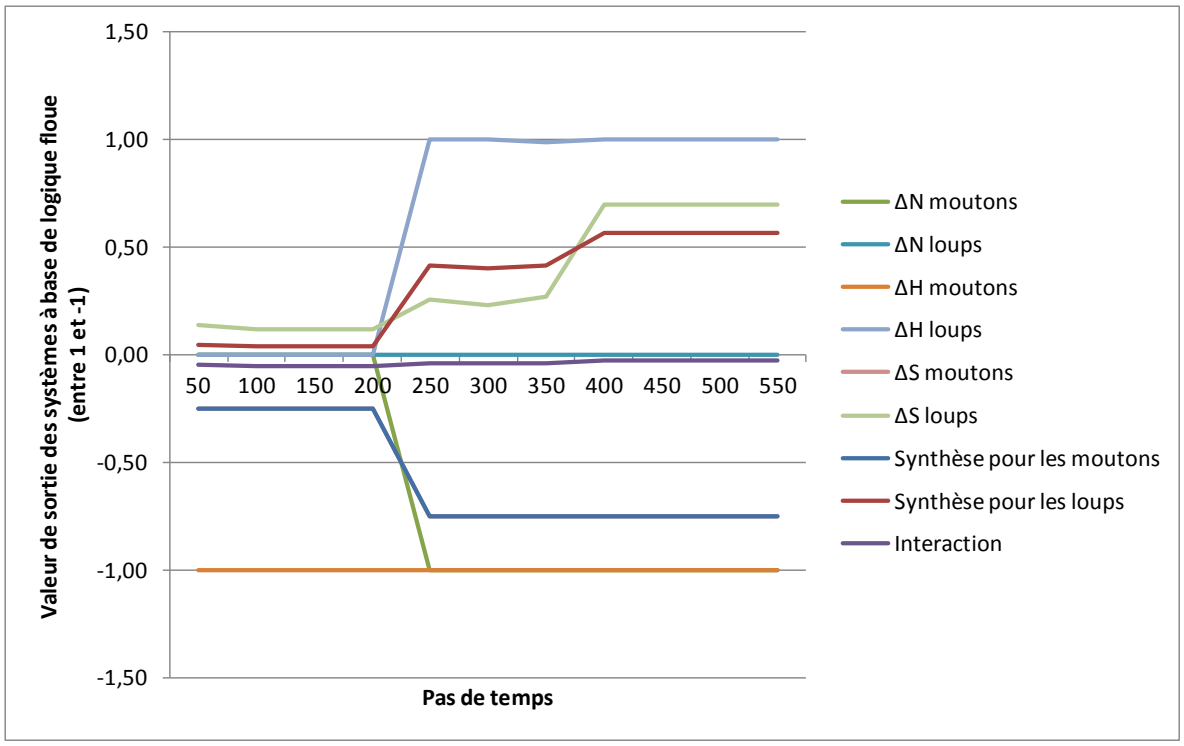


Figure 70 Valeur de l'interaction pour l'exemple du scénario de *Prédation*

Tableau 12 Résultats intermédiaires pour le calcul de l'interaction entre les loups et les moutons

Pas de temps	*Moutons*				*Loups*				Interaction
	ΔN[8]	ΔH[9]	ΔS[10]	Synthèse	ΔN	ΔH	ΔS	Synthèse	
50	0	-1	0	-0,25	0	0	0,13906	0,044241	-0,048271
100	0	-1	0	-0,25	0	0	0,114983	0,037535	-0,050466
150	0	-1	0	-0,25	0	0	0,114983	0,037535	-0,050466
200	0	-1	0	-0,25	0	0	0,114983	0,037535	-0,050466
250	-1	-1	0	-0,75	0	1	0,256626	0,411605	-0,041157

[8] ΔN représente la variation du nombre d'Organismes de l'Espèce (voir section « Calcul du nombre d'Organismes vivants »). Ne possède pas d'unité.

[9] ΔH représente la variation l'Habitat de l'Espèce (section « Calcul de l'Habitat »). Ne possède pas d'unité.

[10] ΔS représente la variation l'ÉtatDeSanté de l'Espèce (section « Calcul de l'état de santé global d'une Espèce »). Ne possède pas d'unité.

300	-1	-1	0	-0,75	0	1	0,229254	0,399668	-0,042326
350	-1	-1	0	-0,75	0	1	0,272603	0,414539	-0,040869
400	-1	-1	0	-0,75	0	1	0,695101	0,568556	-0,02494
450	-1	-1	0	-0,75	0	1	0,695101	0,568556	-0,02494
500	-1	-1	0	-0,75	0	1	0,695101	0,568556	-0,02494
550	-1	-1	0	-0,75	0	1	0,695101	0,568556	-0,02494

Tableau 13 Pas de temps où un *Organisme* meurt en cours de simulation de l'exemple du scénario de préation

Espèce	Pas de temps
Mouton	103
Mouton	106
Mouton	107
Mouton	210
Mouton	214
Mouton	229
Mouton	323
Mouton	383
Loup	407
Loup	520
Loup	580

Tableau 14 Extrait des données brutes de simulation[11]

Pas de temps	Position X	Position Y	Organisme	Niveau d'énergie
381	26.54	22.43	Mouton 1	100
381	27.09	19.76	Loup 10	33.5
382	ND	ND	Mouton 1	0

[11] Pour l'ensemble des données de cette simulation, voir « ANNEXE : Résultats complet d'un scénario de Prédation»

382	26.07	24.66	Loup 10	99.25[12]

En regardant la Figure 70 et les valeurs dans le Tableau 12 pour les résultats de l'exemple, deux éléments sont à noter. D'une part, la valeur numérique de l'interaction détectée à la fin de la simulation correspond à l'interaction *Prédation* comme le montre la Figure 71.

Figure 71 Structure dynamique du scénario de *Prédation*

La détection d'interactions obtenue par le Framework donne le résultat auquel un utilisateur expert pourrait s'attendre en lançant un scénario impliquant des moutons et des loups. C'est effectivement le cas dans la majorité des simulations de ce scénario et ceci sera discuté un peu plus loin dans le document. D'autre part, deux fluctuations importantes de la valeur de l'interaction au cours de la simulation sont à noter. La première se situe au 250ième pas de temps. C'est à ce moment que la valeur de l'interaction passe de *Neutralisme* à *Prédation*. En effet, lors des pas de temps précédents le 250ième pas de temps, un contact entre l'*Habitat* de loups et celui des moutons prend fin. Il est à noter que c'est seulement à fin d'un contact que la valeur de l'interaction est recalculée. C'est donc pour cette raison qu'il est possible de dire qu'il y a eu fin de contact entre le pas de temps 200 et 250, car la valeur de l'interaction change. Lors de la durée du chevauchement entre les deux *Habitats*, les moutons ont perdu six *Organismes* de leur *Espèce*. De plus, les loups se sont nourris et ont refait le plein d'énergie. Donc, le dernier contact a été néfaste pour les moutons et bénéfique pour les loups. C'est pour cette raison que l'interaction détectée à ce stade en est une de *Prédation*. La seconde fluctuation se situe au pas de temps 400. À ce moment, un loup faible est sur le point de mourir réussit à attraper un mouton lors du dernier contact entre les *Habitats* (voir Tableau 14). Ceci permet donc de confirmer que l'interaction entre les deux *Espèces* est clairement bénéfique pour les loups et néfaste pour les moutons.

25 simulations ont été exécutées afin d'observer si la valeur fournie par la détection d'interactions est toujours de type *Prédation*. Ces dernières ont toutes été lancées à la suite. De plus, la position des *Organismes* a été

[12] 100% mois 0.75% pour le coût de déplacement

assignée aléatoirement sans intervention de la part d'un utilisateur. De cette façon, il est possible de constater si le Framework est constant dans ses détections pour un même scénario avec des valeurs initiales différentes et non-connues de l'utilisateur. Le résultat de la détection d'interactions pour chacune de ces simulations est présenté à la Figure 72.

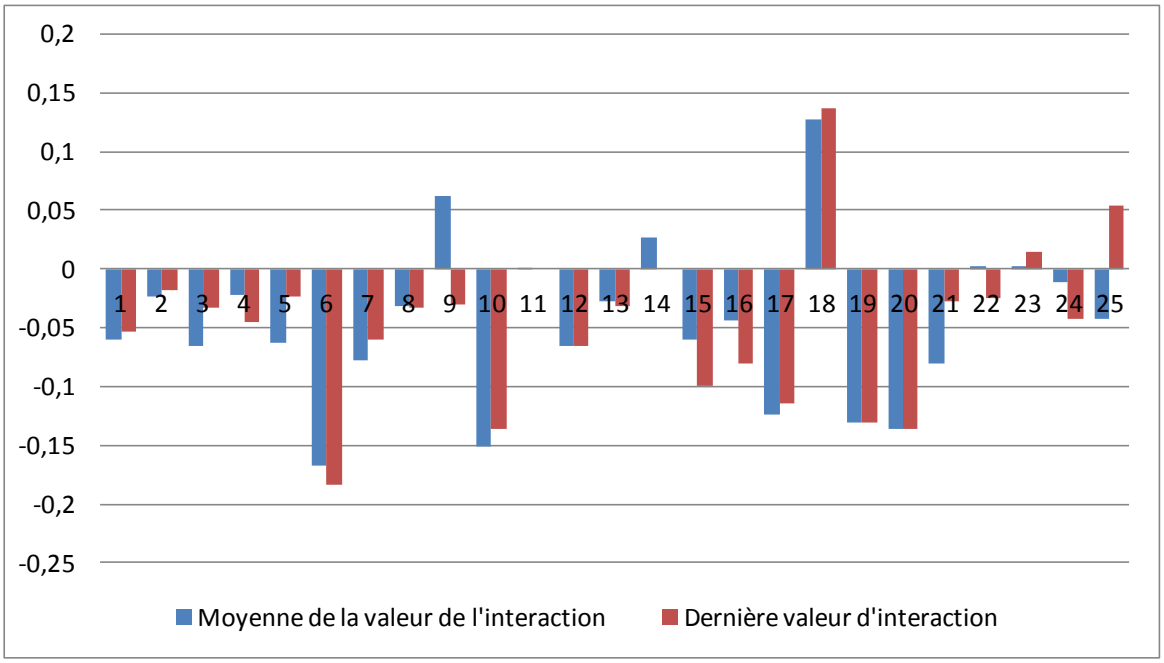


Figure 72 Résultats de la détection d'interactions pour 25 simulations de *Prédation*

Tout d'abord, l'analyse des résultats présentés sur la Figure 72 montre que dans 20 cas (1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 21, 22, 23, 24 et 25), l'interaction détectée est de *Prédation*. Ensuite, dans 4 cas (6, 17, 19 et 20), l'interaction est de type *Amensalisme*. Finalement, dans un cas (18) l'interaction en est une de *Commensalisme*. D'une part, les cas d'*Amensalisme* sont dus au fait que les loups s'épuisent peu à peu jusqu'à leur disparition. En effet, les moutons sont une source d'énergie pour les loups, par contre les moutons ne se laissent pas attraper facilement. C'est donc pour cette raison qu'ultimement, l'apport de l'interaction sur les loups est neutre (bénéfique en nourriture, néfaste en énergie) et que l'interaction est *Amensalisme*. D'autre part, dans le cas où l'interaction détectée est *Commensalisme*, cela est dû à deux facteurs. Le premier est que les loups sont éliminés rapidement, car ils sont incapables d'attraper de moutons pour s'approvisionner. Le second facteur est le grand nombre de moutons restant. En effet, ces derniers ont acquis une grande superficie supplémentaire pour leur *Habitat*, car plus la simulation avance, moins il reste de loups, ce qui implique plus de territoire pour les moutons. Ces simulations montrent donc que l'interaction n'est pas toujours

bénéfique pour les loups et néfaste pour les moutons. Dans certains cas, le système se comporte différemment de ce qui avait été anticipé et ceci est la nature même des systèmes complexes.

En résumé, la moyenne des 25 simulations est de -0.046 ce qui se trouve dans l'intervalle pour la *Prédation* sur la Figure 53. Donc, pour le scénario de prédation entre des loups et des moutons, le Framework détecte une interaction cohérente avec les données de simulation.

Scénario d'invasion de zombies

Cette section présente les résultats et leur analyse pour le scénario d'invasion de zombies présenté à la section « Scénario d'invasion de zombies ». Dans le cas présent, les résultats d'une simulation typique sont présentés.

Les paramètres initiaux pour la génération des résultats de la simulation qui sera présentée sont inscrits au Tableau 15.

Tableau 15 Paramètres initiaux pour les simulations du scénario d'invasion de *Zombies*

Écosystème	
Dimension (unité)	50x50
Pour toutes les Espèces	
Profondeur du champ de vue (unité)	3
Angle de vision	217˚
Zombie	
Nombre	1
Chance de succès lors d'un affrontement avec un *Militaire*	20%

Vitesse (unités/pas de temps)	0.8
Humain	
Nombre	10
Vitesse (unités/pas de temps)	1
Durée de panique (pas de temps)	20
Militaire	
Nombre	3
Chance de succès lors d'un affrontement avec un *Zombie*	80%
Vitesse (unités/pas de temps)	1

Lors de la simulation d'un scénario d'invasion de zombie, trois interactions sont à détecter et répertorier. La première interaction est entre les *Humains* et les *Militaires*. La tendance de la valeur de l'interaction se trouve à la Figure 73. De plus, les valeurs intermédiaires utilisées dans le calcul de l'interaction sont présentées dans le Tableau 16.

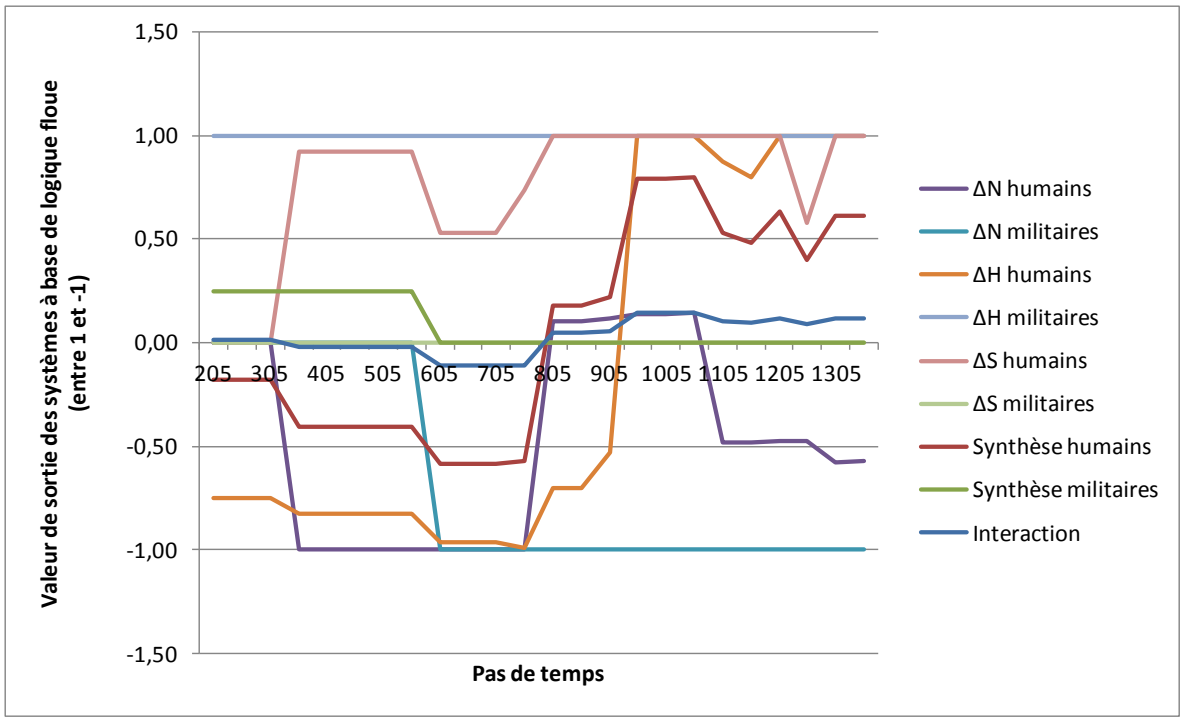


Figure 73 Valeur de l'interaction entre les *Humains* et les *Militaires*

Tableau 16 Résultats intermédiaires pour le calcul de l'interaction entre les *Humains* et les *Militaires*

Pas de temps	*Humains*				*Militaires*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	
205	0,00	-0,75	0,00	-0,18	0,00	1,00	0,00	0,25	0,01
255	0,00	-0,75	0,00	-0,18	0,00	1,00	0,00	0,25	0,01
305	0,00	-0,75	0,00	-0,18	0,00	1,00	0,00	0,25	0,01
355	-1,00	-0,82	0,92	-0,40	0,00	1,00	0,00	0,25	-0,02
405	-1,00	-0,82	0,92	-0,40	0,00	1,00	0,00	0,25	-0,02
455	-1,00	-0,82	0,92	-0,40	0,00	1,00	0,00	0,25	-0,02
505	-1,00	-0,82	0,92	-0,40	0,00	1,00	0,00	0,25	-0,02
555	-1,00	-0,82	0,92	-0,40	0,00	1,00	0,00	0,25	-0,02
605	-1,00	-0,96	0,53	-0,59	-1,00	1,00	0,00	0,00	-0,11
655	-1,00	-0,96	0,53	-0,59	-1,00	1,00	0,00	0,00	-0,11
705	-1,00	-0,96	0,53	-0,59	-1,00	1,00	0,00	0,00	-0,11
755	-1,00	-0,99	0,73	-0,57	-1,00	1,00	0,00	0,00	-0,11
805	0,11	-0,70	1,00	0,18	-1,00	1,00	0,00	0,00	0,05
855	0,11	-0,70	1,00	0,18	-1,00	1,00	0,00	0,00	0,05
905	0,12	-0,53	1,00	0,22	-1,00	1,00	0,00	0,00	0,06

955	0,14	1,00	1,00	0,79	-1,00	1,00	0,00	0,00	0,14
1005	0,14	1,00	1,00	0,79	-1,00	1,00	0,00	0,00	0,14
1055	0,14	1,00	1,00	0,80	-1,00	1,00	0,00	0,00	0,14
1105	-0,48	0,87	1,00	0,53	-1,00	1,00	0,00	0,00	0,10
1155	-0,48	0,80	1,00	0,48	-1,00	1,00	0,00	0,00	0,10
1205	-0,48	1,00	1,00	0,63	-1,00	1,00	0,00	0,00	0,12
1255	-0,47	1,00	0,58	0,40	-1,00	1,00	0,00	0,00	0,09
1305	-0,58	1,00	1,00	0,61	-1,00	1,00	0,00	0,00	0,11
1355	-0,57	1,00	1,00	0,61	-1,00	1,00	0,00	0,00	0,11

La Figure 73 présente en rouge la valeur de la synthèse pour les Humains. La courbe en vert montre la valeur de la synthèse pour les Militaires. La courbe en bleu présente la valeur de l'interaction.

Par l'analyse de la Figure 73 et du Tableau 16, trois fluctuations sont à noter dans la valeur de l'interaction entre les *Humains* et les *Militaires*. La première se produit au pas de temps 605. C'est à cet instant que l'interaction passe du *Neutralisme* à l'*Amensalisme*. En effet, lors du contact précédant ce changement, les *Militaires* ont perdu un *Organisme* qui a été converti en *Zombie*. Ce qui implique que la synthèse pour les *Militaires* a été influencée négativement. De plus, les *Humains* se sont vu infliger des pertes en énergie et des pertes d'*Organismes* causées par la présence de *Zombies* dans leur entourage. Donc, la valeur de l'interaction entre les *Humains* et *Militaires* a été influencée par la présence de *Zombies* lors du dernier contact entre les deux *Habitats*. La seconde fluctuation de la valeur de l'interaction se produit au pas de temps 805. C'est à ce moment que l'interaction retourne au *Neutralisme*. En effet, le dernier contact entre les *Humains* et les *Militaires* a permis aux *Humains* de reprendre des forces, car les *Militaires* ont éliminé les *Zombies*. Par le fait même, les *Humains* se sont vu protéger par l'action des *Militaires*. Donc, les *Militaires* commencent à avoir une influence positive sur les *Humains*. Le dernier changement de valeur de l'interaction se produit au pas de temps 955. C'est à cet instant que débute la convergence vers une valeur finale d'interaction pour le scénario analysé. En effet, le dernier contact à permis aux *Humains* de s'épanouir en augmentant la dimension de leur *Habitat*. Les *Militaires* ont définitivement assumé leur rôle de protection indirect pour les *Humains* en éliminant les *Zombies*. L'interaction entre les *Humains* et les *Militaires* est ultimement une interaction de *Commensalisme*. C'est précisément ce à quoi l'utilisateur pouvait s'attendre. C'est-à-dire que l'interaction est bénéfique pour les *Humains* mais *Neutre* pour les *Militaires* qui ne soutirent rien de ces derniers en échange de protection.

La deuxième interaction à évaluer est entre les *Humains* et les *Zombies*. La fluctuation de la valeur de cette dernière se trouve à la Figure 74 et les valeurs intermédiaires utilisées dans son calcul sont présentées dans le Tableau 17.

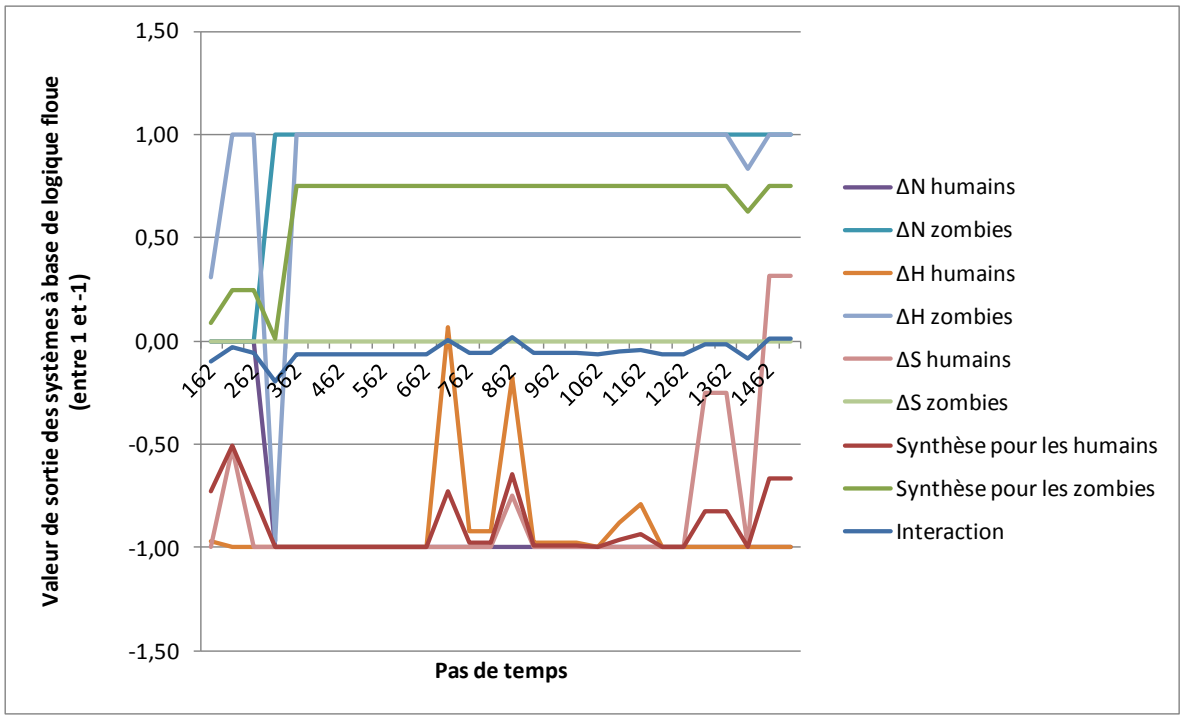


Figure 74 Valeur de l'interaction entre les *Humains* et les *Zombies*

Tableau 17 Résultats intermédiaires pour le calcul de l'interaction entre les *Humains* et les *Zombies*

Pas de temps	*Humains*				*Zombies*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	
162	0,00	-0,97	-1,00	-0,73	0,00	0,31	0,00	0,09	-0,10
212	0,00	-1,00	-0,52	-0,51	0,00	1,00	0,00	0,25	-0,03
262	0,00	-1,00	-1,00	-0,75	0,00	1,00	0,00	0,25	-0,06
312	-1,00	-1,00	-1,00	-1,00	1,00	-0,97	0,00	0,01	-0,20
362	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
412	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
462	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
512	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06

562	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
612	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
662	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
712	-1,00	0,06	-1,00	-0,73	1,00	1,00	0,00	0,75	0,00
762	-1,00	-0,92	-1,00	-0,97	1,00	1,00	0,00	0,75	-0,05
812	-1,00	-0,92	-1,00	-0,97	1,00	1,00	0,00	0,75	-0,05
862	-1,00	-0,17	-0,75	-0,65	1,00	1,00	0,00	0,75	0,02
912	-1,00	-0,98	-1,00	-0,99	1,00	1,00	0,00	0,75	-0,06
962	-1,00	-0,98	-1,00	-0,99	1,00	1,00	0,00	0,75	-0,06
1012	-1,00	-0,98	-1,00	-0,99	1,00	1,00	0,00	0,75	-0,06
1062	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
1112	-1,00	-0,88	-1,00	-0,96	1,00	1,00	0,00	0,75	-0,05
1162	-1,00	-0,79	-1,00	-0,94	1,00	1,00	0,00	0,75	-0,04
1212	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
1262	-1,00	-1,00	-1,00	-1,00	1,00	1,00	0,00	0,75	-0,06
1312	-1,00	-1,00	-0,25	-0,82	1,00	1,00	0,00	0,75	-0,01
1362	-1,00	-1,00	-0,25	-0,82	1,00	1,00	0,00	0,75	-0,01
1412	-1,00	-1,00	-1,00	-1,00	1,00	0,83	0,00	0,63	-0,08
1462	-1,00	-1,00	0,32	-0,66	1,00	1,00	0,00	0,75	0,01
1512	-1,00	-1,00	0,32	-0,66	1,00	1,00	0,00	0,75	0,01

La Figure 75 présente la valeur de l'interaction en bleu. Les deux valeurs intermédiaires de synthèse pour les Humains et les Zombies sont respectivement en rouge et en vert.

L'analyse de la Figure 75 et du Tableau 17 montre deux changements dans la valeur de l'interaction entre les *Humains* et *Zombies*. Le premier changement se situe au pas de temps 312. C'est à ce moment que l'interaction passe de *Prédation* à *Amensalisme*. Ce changement est dû à une diminution de la dimension de l'*Habitat* des *Zombies*. Ce phénomène se produit parfois lorsque les déplacements des *Organismes* sont aléatoires. En effet, ils peuvent subitement se diriger tous au même moment vers le centre de l'*Habitat* ce qui engendre une diminution de la taille ce dernier. Le second changement se situe au pas de temps 362. À ce stade, l'interaction revient à la *Prédation*. En effet, la fluctuation négative de l'*Habitat* des *Zombies* notée précédemment n'était que passagère. Donc l'interaction converge vers la valeur à laquelle l'utilisateur pouvait s'attendre initialement, c'est-à-dire, une interaction *Néfaste* pour les *Humains* mais bénéfique pour les *Zombies*.

La dernière interaction à évaluer pour ce scénario est entre les *Militaires* et les *Zombies*. La tendance de la valeur de cette dernière se trouve à la Figure 75 et les valeurs intermédiaires utilisées pour son calcul sont présentées au Tableau 18.

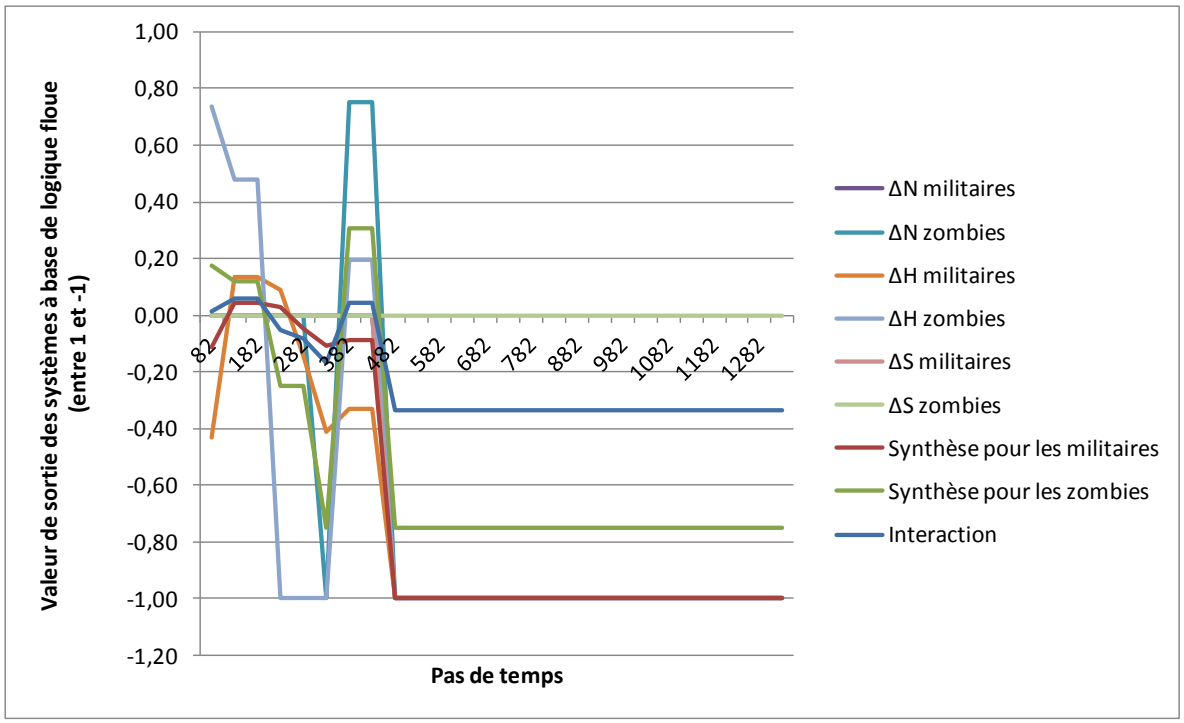


Figure 75 Valeur de l'interaction entre les *Militaires* et les *Zombies*

Tableau 18 Résultats intermédiaires pour le calcul de l'interaction entre les *Militaires* et les *Zombies*

Pas de temps	*Militaires*				*Zombies*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	
82	0,00	-0,43	0,00	-0,11	0,00	0,74	0,00	0,18	0,01
132	0,00	0,14	0,00	0,04	0,00	0,48	0,00	0,12	0,06
182	0,00	0,14	0,00	0,04	0,00	0,48	0,00	0,12	0,06
232	0,00	0,09	0,00	0,03	0,00	-1,00	0,00	-0,25	-0,05
282	0,00	-0,14	0,00	-0,05	0,00	-1,00	0,00	-0,25	-0,08
332	0,00	-0,41	0,00	-0,11	-1,00	-1,00	0,00	-0,75	-0,17
382	0,00	-0,33	0,00	-0,09	0,75	0,19	0,00	0,31	0,04
432	0,00	-0,33	0,00	-0,09	0,75	0,19	0,00	0,31	0,04

482	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
532	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
582	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
632	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
682	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
732	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
782	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
832	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
882	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
932	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
982	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1032	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1082	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1132	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1182	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1232	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1282	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34
1332	-1,00	-1,00	-1,00	-1,00	-1,00	-1,00	0,00	-0,75	-0,34

La Figure 75 présente la valeur de l'interaction en bleu. Les deux valeurs de synthèse pour les *Militaires* et les *Zombies* sont respectivement en rouge et en vert.

Le Figure 75 et le Tableau 18 permettent d'observer trois changements dans la valeur de l'interaction au cours de la simulation. Le premier changement se trouve au pas de temps 332. En effet, depuis le début de la simulation la synthèse pour les *Zombies* se dégrade peu à peu jusqu'au pas de temps 332 où l'interaction passe de *Neutralisme* à *Amensalisme*. En effet, c'est lors du dernier contact entre les *Habitats* que les *Zombies* se sont vu infliger leurs premières pertes d'*Organismes*. Le deuxième changement si situe au pas de temps 382. C'est à cet instant que les *Zombies* ayant subi plusieurs pertes précédemment, reprennent une partie de l'*Habitat* perdu. C'est donc pour cette raison qu'un gain est noté au niveau de la synthèse des *Zombies* et que l'interaction repasse aux *Neutralisme*. Le dernier changement se situe au pas de temps 482. C'est à ce moment que la valeur de l'interaction converge vers la *Compétition*. Lors du dernier contact entre les *Habitats*, les *Militaires* ont perdu un premier *Organisme* auprès des *Zombies*. Désormais, l'interaction est néfaste pour les deux Espèces, car chacune d'entre elles tente d'éliminer l'autre.

En résumé, les trois interactions anticipées ont été détectées et sont présentées à la Figure 76.

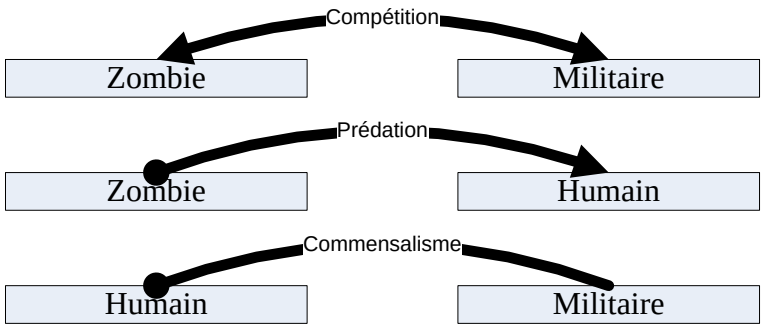


Figure 76 Structure dynamique du scénario de l'invasion de *Zombies*

La première interaction qui implique les *Humains* et les *Militaires* a été identifiée comme étant une interaction de *Commensalisme* ce qui correspond tout à fait à ce à quoi l'utilisateur peut s'attendre. En effet, l'élimination des *Zombies* par les *Militaires* s'avère bénéfique pour les *Humains*. Ce qui implique que lorsqu'un *Militaire* se trouve à proximité d'un *Humain*, ce dernier se voit protégé en partie des agressions perpétuées par les *Zombies*. À l'opposé, le *Militaire* n'obtient rien de concret de l'interaction, car il ne retire rien de son interaction avec les *Humains*. La deuxième interaction détectée est entre les *Humains* et les *Zombies*. Cette fois-ci la détection a identifié l'interaction comme étant de la *Prédation*. Une fois de plus cette détection est cohérente. En effet, le *Zombies* a comme seul objectif d'infecter les autres *Organismes* présents dans l'*Écosystème*. Donc, les *Humains* se font chasser par les *Zombies* ce qui leur est néfaste. Finalement, la dernière interaction entre les *Militaires* et les *Zombies* est de type *Compétition*. Dans les deux cas, la première *Espèce* tente d'éliminer la seconde. L'utilisateur observe une compétition entre les *Organismes* de chacune des deux *Espèces* pour leur survie ce qui est en concordance avec les attentes pour un tel scénario.

Scénario de bactéries

Le dernier scénario démontrera l'efficacité du Framework ainsi que son algorithme de détection d'interactions lors d'une simulation interactive. En effet, le scénario de bactéries permet à un utilisateur de modifier le scénario en cours de simulation. La validité de l'approche proposée sera présentée encore une fois à l'aide d'un exemple.

Les paramètres initiaux pour la génération des résultats de la simulation qui sera présentée sont listés dans le Tableau 19.

Tableau 19 Paramètres initiaux pour les simulations du scénario d'invasion de *Zombies*

Écosystème	
Dimension (unités)	30x30
Pour toutes les Espèces	
Profondeur du champ de vue (unités)	1
Angle de vision	360°
Coût énergétique de la division cellulaire	20% de l'énergie restante
Roche	
Nombre	4
Papier	
Nombre	4
Ciseau	
Nombre	4 (ajoutés au pas de temps 590 dans le cas présenté)

Lors de l'exécution du scénario sur l'évolution de bactéries, trois interactions sont à détecter par le Framework. De plus, comme il a été montré à la section « Scénario d'évolution de bactéries », lors de la simulation d'un tel scénario, les trois interactions devraient normalement toutes être d'*Amensalisme*. En effet, chaque *Espèce* influence négativement une autre sans obtenir de bénéfice.

La première interaction est entre *Roche* et *Papier*. Sa valeur est représentée sur la Figure 77.

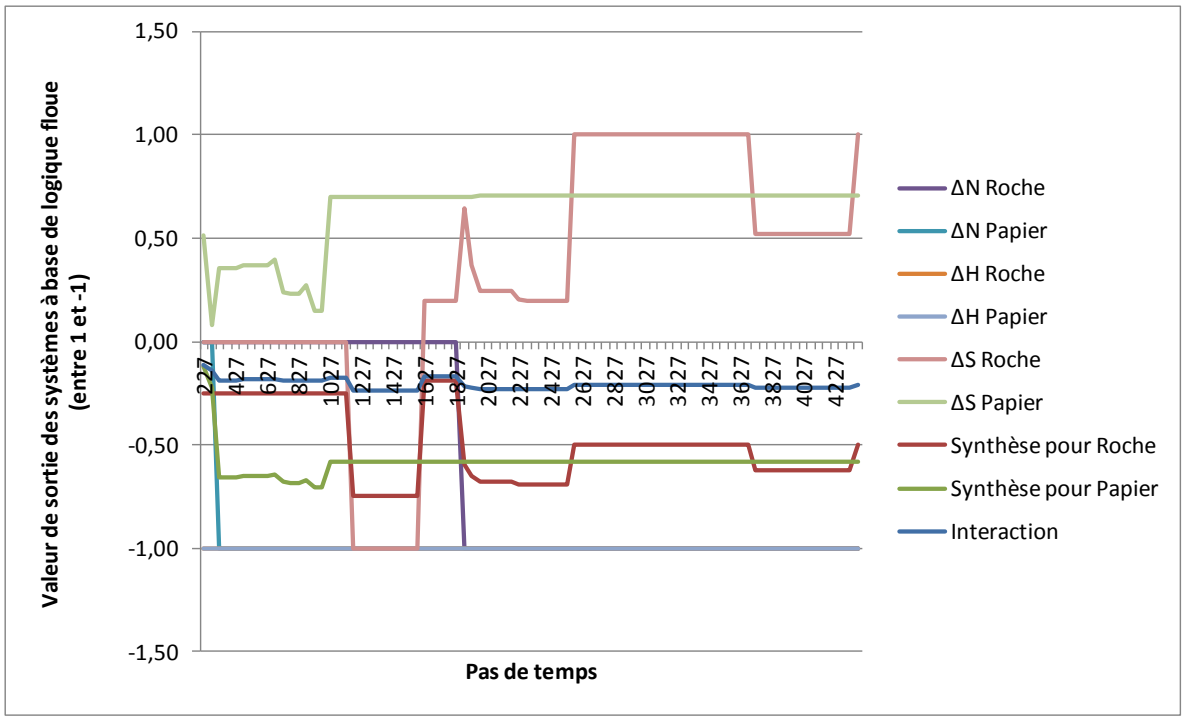


Figure 77 Valeur de l'interaction entre les *Roches* et les *Papiers*

Cette figure présente trois courbes. La valeur de l'interaction est présentée en bleu. La valeur de la synthèse de l'interaction pour les *Roches* et les *Papiers* sont respectivement présentées en rouge et vert. La valeur détaillée des courbes pour chaque pas de temps ainsi que les valeurs intermédiaires de calculs sont présentées au Tableau 20. Il est à noter que les valeurs d'interaction répétitives ont été enlevées afin d'alléger le tableau.

Tableau 20 Résultats intermédiaires pour le calcul de l'interaction entre les *Roches* et les *Papiers*

Pas de temps	*Roche*				*Papier*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	
227	0,00	-1,00	0,00	-0,25	0,00	-1,00	0,52	-0,12	-0,11
277	0,00	-1,00	0,00	-0,25	0,00	-1,00	0,08	-0,22	-0,14
327	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,35	-0,65	-0,19
627	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,37	-0,65	-0,19
677	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,40	-0,65	-0,18
727	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,24	-0,68	-0,19
777	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,23	-0,68	-0,19
827	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,23	-0,68	-0,19

877	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,28	-0,67	-0,19
927	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,15	-0,70	-0,19
977	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,15	-0,70	-0,19
1027	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,70	-0,58	-0,18
1077	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,70	-0,58	-0,18
1127	0,00	-1,00	0,00	-0,25	-1,00	-1,00	0,70	-0,58	-0,18
1177	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1227	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1277	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1327	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1377	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1427	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1477	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1527	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1577	0,00	-1,00	-1,00	-0,75	-1,00	-1,00	0,70	-0,58	-0,24
1627	0,00	-1,00	0,19	-0,19	-1,00	-1,00	0,70	-0,58	-0,17
1877	-1,00	-1,00	0,64	-0,60	-1,00	-1,00	0,70	-0,58	-0,22
1927	-1,00	-1,00	0,37	-0,65	-1,00	-1,00	0,70	-0,58	-0,22
1977	-1,00	-1,00	0,25	-0,68	-1,00	-1,00	0,70	-0,58	-0,23
2577	-1,00	-1,00	1,00	-0,50	-1,00	-1,00	0,70	-0,58	-0,21
3727	-1,00	-1,00	0,52	-0,62	-1,00	-1,00	0,70	-0,58	-0,22
4377	-1,00	-1,00	1,00	-0,50	-1,00	-1,00	0,70	-0,58	-0,21

À priori, dès la fin du premier contact entre les deux *Espèces* (au pas de temps 227), la valeur de l'interaction détectée est d'*Amensalisme* comme présenté à la Figure 78.

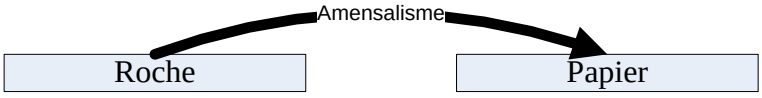


Figure 78 Structure dynamique entre *Roche* et *Papier* avant l'insertion de *Ciseau*

Tel que mentionné à la section « Scénario d'évolution de bactéries », avant l'insertion de la bactérie *Ciseau* dans l'*Écosystème*, la *Roche* est néfaste au *Papier*. C'est au pas de temps 1177 que l'interaction passe à l'*Amensalisme* car c'est à ce moment que se font sentir pour la première fois les effets de l'insertion du *Ciseau* sur l'interaction entre le *Papier* et la *Roche*. C'est désormais le *Papier* qui est néfaste à la *Roche* comme le montre la Figure 79.

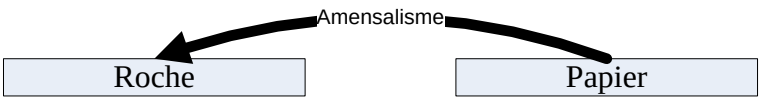


Figure 79 Structure dynamique entre *Roche* et *Papier* après l'insertion de *Ciseau*

La valeur de l'interaction identifiée reflète ce phénomène. Bien entendu la valeur même de l'interaction ne varie pas énormément dû au fait qu'elle demeure tout de même de l'*Amensalisme*. Par contre, les valeurs de synthèse pour les deux *Espèces* montrent très clairement ce phénomène. En effet, au pas de temps 1177 la valeur de synthèse passe de -0.25 à -0.75 dû à la diminution de l'état de santé des *Organismes* de l'*Espèces Roche*. La valeur de synthèse du *Papier* reste tout de même négative à cause d'un phénomène de mémoire, car, initialement, c'était le *Papier* qui était influencé négativement pas l'interaction. Cette valeur diminue cependant peu à peu à partir de l'insertion du *Ciseau*. Finalement, au pas de temps 1627 certaines oscillations de la valeur de synthèse sont à noter et ce, jusqu'à la fin de la simulation. Ces dernières sont causées par les *Roches* qui se trouvent à l'intérieur de l'*Habitat* des *Papiers* sans être trop près de ces derniers. De plus, les *Roches* ont trouvé les derniers nutriments de l'*Écosystème* à ne pas avoir été consommés ce qui leur permet de reprendre de l'énergie. Par contre, l'interaction reste tout de même une interaction d'*Amensalisme* comme les organigrammes des *Espèces* l'impliquaient.

La Figure 80 et la Tableau 21 présentent la valeur de l'interaction entre les *Roches* et les *Ciseaux*.

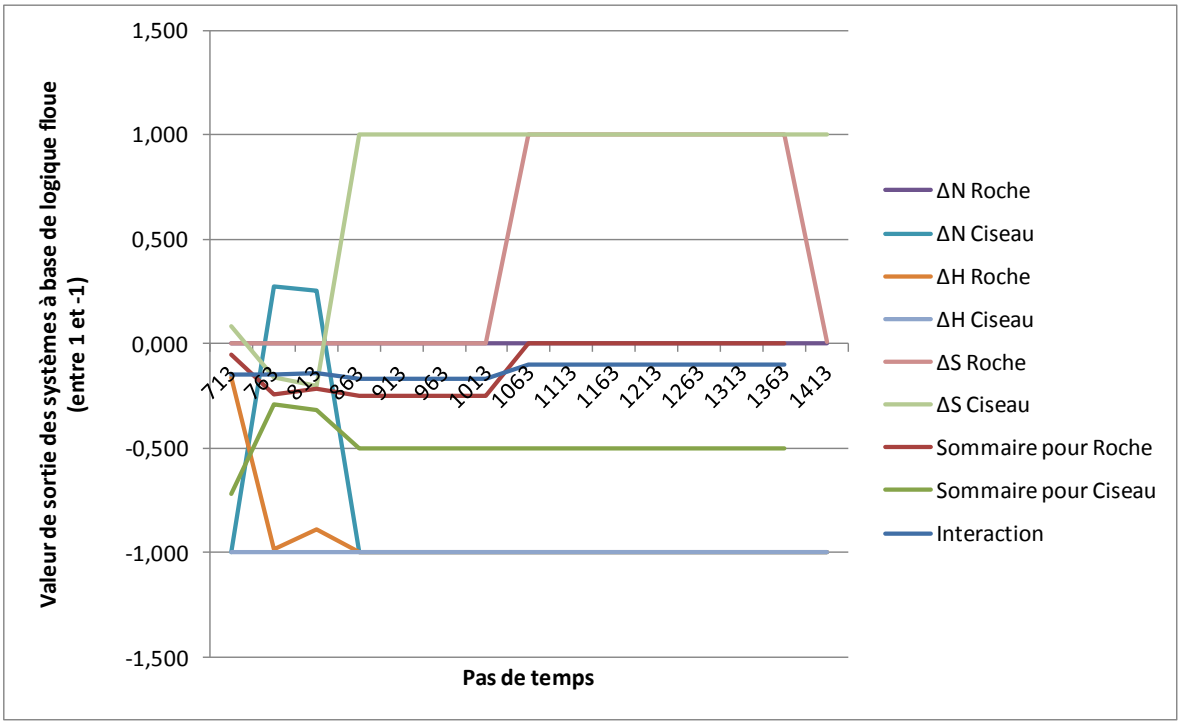


Figure 80 Valeur de l'interaction entre les *Roches* et les *Ciseaux*

Tableau 21 Résultats intermédiaires pour le calcul de l'interaction entre les *Roches* et les *Ciseaux*

Pas de temps	*Roche*				*Ciseau*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	
713	0,00	-0,16	0,00	-0,05	-1,00	-1,00	0,08	-0,72	-0,15
763	0,00	-0,98	0,00	-0,24	0,27	-1,00	-0,16	-0,29	-0,15
813	0,00	-0,89	0,00	-0,21	0,26	-1,00	-0,20	-0,31	-0,14
863	0,00	-1,00	0,00	-0,25	-1,00	-1,00	1,00	-0,50	-0,17
913	0,00	-1,00	0,00	-0,25	-1,00	-1,00	1,00	-0,50	-0,17
963	0,00	-1,00	0,00	-0,25	-1,00	-1,00	1,00	-0,50	-0,17
1013	0,00	-1,00	0,00	-0,25	-1,00	-1,00	1,00	-0,50	-0,17
1063	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1113	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1163	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1213	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1263	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1313	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10
1363	0,00	-1,00	1,00	0,00	-1,00	-1,00	1,00	-0,50	-0,10

Sur la Figure 80 la valeur de l'interaction est représentée en bleu. Les valeurs de synthèses pour les *Roches* et les *Ciseaux* sont respectivement en rouge et en vert. Deux pas de temps sont à noter pour l'interaction entre ces deux *Espèces*. Dès le tout début, l'interaction détectée est une interaction d'*Amensalisme*. Par contre, ceci est dû au fait qu'elle est aussi néfaste pour les *Roches*. En effet, ce ne sont pas les *Ciseaux* qui sont directement néfastes pour les *Roches*. C'est plutôt lié au fait que l'ajout des *Ciseaux* rend désormais le *Papier* néfaste pour la *Roche*. Donc, une période d'oscillation est à prévoir pour la valeur de synthèse pour la *Roche* qui doit s'adapter au changement de comportement de *Papier*. C'est au pas de temps 1063 que l'interaction n'est plus néfaste pour la *Roche* et que la valeur de l'interaction converge vers *Amensalisme*.

La dernière interaction à analyser est celle entre le *Papier* et le *Ciseau* qui est résumée par la Figure 81 et le Tableau 22.

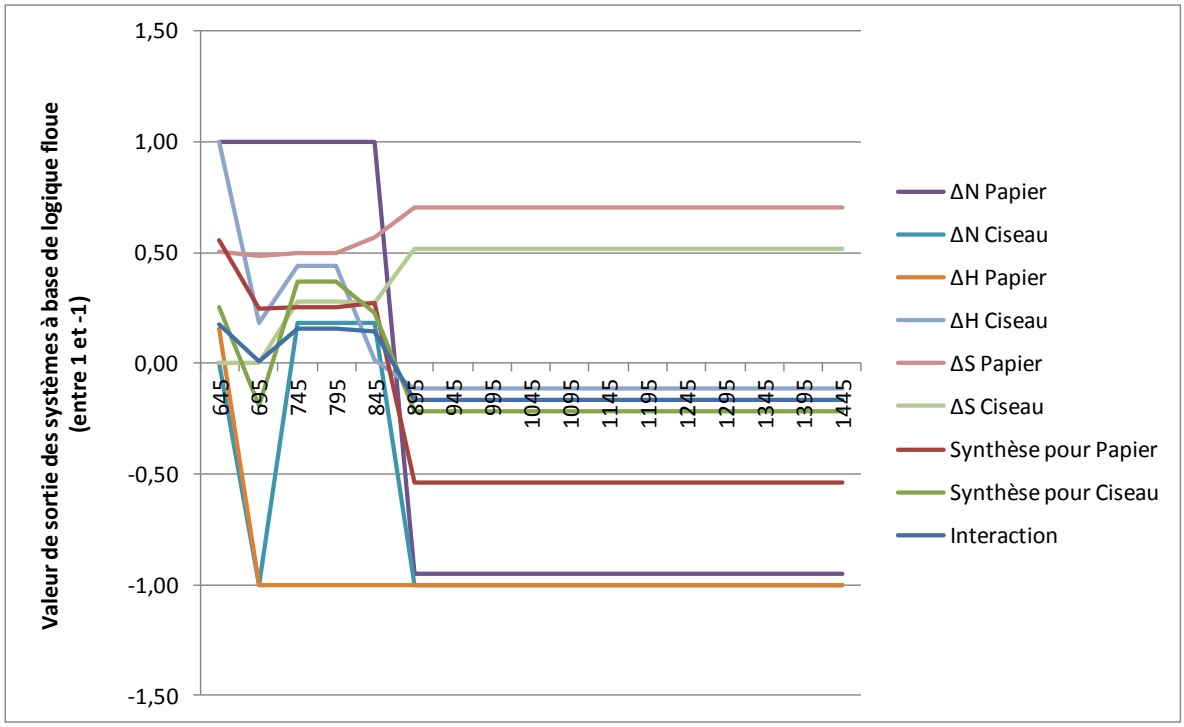


Figure 81 Valeur de l'interaction entre les *Papiers* et les *Ciseaux*

Tableau 22 Résultats intermédiaires pour le calcul de l'interaction entre les *Papiers* et les *Ciseaux*

Pas de temps	*Papier*				*Ciseau*				Interaction
	ΔN	ΔH	ΔS	Synthèse	ΔN	ΔH	ΔS	Synthèse	

645	1,00	0,15	0,51	0,56	0,00	1,00	0,00	0,25	0,17
695	1,00	-1,00	0,49	0,25	-1,00	0,18	0,00	-0,19	0,01
745	1,00	-1,00	0,50	0,25	0,19	0,44	0,28	0,37	0,15
795	1,00	-1,00	0,50	0,25	0,19	0,44	0,28	0,37	0,15
845	1,00	-1,00	0,57	0,27	0,18	0,01	0,27	0,23	0,14
895	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
945	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
995	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1045	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1095	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1145	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1195	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1245	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1295	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1345	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1395	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17
1445	-0,95	-1,00	0,70	-0,54	-1,00	-0,11	0,52	-0,22	-0,17

Deux pas de temps sont à noter suite à l'analyse de la figure et du tableau. Le premier se trouve au pas de temps 645 soit la première fin de contact entre les deux *Espèces*. À cet instant, l'interaction est de type *Commensalisme*. La modification du comportement du *Papier* envers la *Roche* depuis l'insertion des *Ciseaux* engendre ce phénomène. Le *Papier* n'est désormais plus affecté négativement par les *Roches* et un regain d'énergie et une diminution du taux de mortalité sont observés. Par contre, ce phénomène n'est que temporaire. En effet, au pas de temps 895, les *Ciseaux* ont pris leur place dans l'*Écosystème* et ils sont néfastes au le *Papier*. L'interaction passe à l'*Amensalisme* après avoir brièvement passé du *Commensalisme* au *Neutralisme*.

En résumé, on peut voir que les résultats pour chaque interaction mènent au résultat final attendu pour un tel scénario. La Figure 82 présente la valeur de chacune d'entre elles.

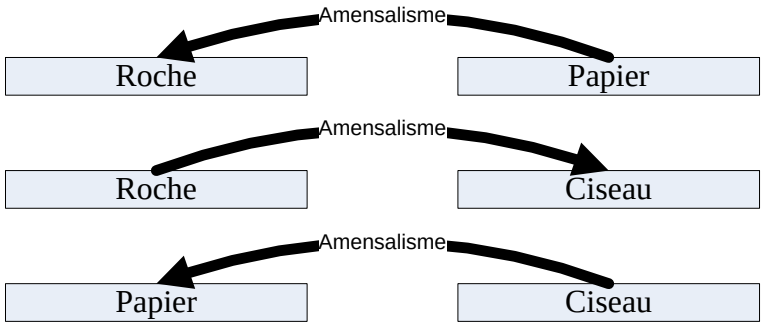


Figure 82 Structure dynamique après l'insertion de *Ciseau*

Chaque *Espèce* du scénario d'évolution de *Bactéries* est affectée négativement par une autre *Espèce*. Donc, les trois interactions sont de type *Amensalisme*.

Conclusion

Ce chapitre fera tout d'abord un retour sur les chapitres précédents. Ensuite, les contributions de la thèse seront présentées. Finalement, les perspectives d'avenir pour le Framework seront introduites.

Retour sur les chapitres précédents

Le chapitre sur l'état de l'art montre qu'il n'y a pas de travaux qui se sont attardés directement au support automatique à la simulation visuelle interactive. En effet, les études sur le sujet se concentrent davantage sur la façon de présenter l'information disponible en cours de simulation que sur la génération de nouvelle information. C'est pour cette raison que le projet de thèse s'est concentré sur cette question. Pour palier au manquement d'assistance automatique en cours de simulation visuelle interactive, plusieurs autres domaines de recherche ont été explorés afin de s'inspirer de leur expérience. Tout d'abord, le domaine de la simulation a été abordé afin d'être en mesure de dresser le portrait actuel. Ensuite, un survol du champ d'étude sur la simulation visuelle interactive a permis de cibler les avancements pouvant être exploités dans nos travaux. De plus, l'étude de ce domaine a permis de bien cadrer les travaux de recherche abordés dans la thèse. Par la suite, les caractéristiques des systèmes complexes ont été détaillées. Grâce à cette analogie, il a été possible de conclure qu'il y a place à l'amélioration afin de faciliter la compréhension des systèmes complexes simulés. Ensuite, plusieurs modèles conceptuels ont été étudiés ce qui a permis de développer un nouveau modèle conceptuel pour le projet de thèse qui s'adapte à une multitude de scénarios de simulation. Finalement, le domaine de l'intelligence artificielle a été exploré afin d'identifier des techniques efficaces de détection d'interaction. Les arbres de décisions, les réseaux de neurones et la logique floue sont des techniques qui ont été évaluées de façon à identifier laquelle répondrait le mieux aux besoins de détection automatique d'interactions et de l'identification de celles-ci.

Le Chapitre 3 se concentre sur l'architecture du Framework qui se base sur un modèle conceptuel développé précisément pour les besoins du projet de thèse. D'une part, le modèle conceptuel mis au point s'inspire largement du paradigme des écosystèmes. En effet, le vocabulaire accompagnant le projet de thèse emprunte plusieurs termes provenant de l'écologie. De plus, les structures statique et dynamique du Framework s'inspirent des structures statique et dynamique des écosystèmes naturels. De cette façon, l'utilisateur est en mesure de se familiariser aisément avec le Framework et les résultats qu'il génère, car la grande majorité des termes sont d'usage courant.

Le Chapitre 4 aborde en détail les algorithmes qui implantent le Framework et qui permettent de faciliter le travail de compréhension de la part de l'utilisateur de la simulation. Pour ce faire, le chapitre est divisé en deux parties soit les calculs préliminaires et les algorithmes de détection d'interactions. D'une part, les calculs préliminaires représentent la première étape dans le processus de détection et d'identification automatique des interactions entre les entités simulées. Trois algorithmes ont été développés afin de calculer de nouvelles propriétés qui seront à la base des calculs subséquents. Les trois algorithmes permettent de calculer la dimension des habitats, l'état de santé global des *Espèces* et le nombre d'*Organismes* vivants pour chaque *Espèce*. La seconde et dernière étape de calculs préliminaires consiste à la quantification des propriétés calculées à l'étape précédente. Cette phase a comme objectif de ramener la valeur des propriétés calculées entre -1 et 1. Pour ce faire, l'algorithme compare les valeurs des propriétés calculées lorsqu'il y a contact entre les *Habitats* des *Espèces* (possibilité d'interaction) et lorsqu'il n'y a pas de contact. De cette façon, il est possible de savoir si un contact est néfaste, neutre ou bénéfique pour chacune des trois propriétés calculées et ce, pour chaque *Espèce*. D'autre part, une fois connu l'apport de l'interaction sur chaque propriété calculée, il est possible de déduire le type d'interaction en cause entre chaque paire d'*Espèces*. Ceci se fait en deux étapes soit la synthèse des valeurs quantifiées et l'identification de l'interaction. D'une part, la synthèse des valeurs quantifiées consiste tout simplement à utiliser la quantification des trois propriétés calculées afin de résumer le type d'influence que chaque *Espèce* subit (néfaste, neutre ou bénéfique). D'autre part, les valeurs de synthèse pour chaque paire d'*Espèce* permettent d'identifier le type d'interaction présente dans le système simulé.

Le Chapitre 5 présente les résultats obtenus lorsque le Framework est utilisé pour des simulations visuelles interactives de systèmes complexes. En premier lieu, trois scénarios avec des objectifs bien précis sont détaillés. Le premier scénario simule le phénomène de prédation entre les moutons et les loups. L'objectif de ce scénario est de montrer que le Framework reproduit les mêmes résultats pour un même scénario lancé à plusieurs reprises avec des paramètres initiaux différents. Le deuxième scénario consiste à simuler une invasion de zombies au sein d'une population de civils et de militaires. Son but est de valider les résultats fournis par le Framework dans un cas complexe. Le dernier scénario valide le Framework dans le cas où il y a une intervention de la part d'un utilisateur en cours de simulation. Dans ce cas, l'objectif est de valider la détection d'interaction lors d'une simulation visuelle interactive.

En second lieu, les résultats obtenus avec les trois scénarios sont présentés et analysés. Tout d'abord, les résultats du scénario de prédation montrent que le Framework donne des valeurs d'interactions constantes même si les valeurs initiales des propriétés diffèrent pour chaque simulation d'un même scénario. Ceci

implique que le Framework est en mesure de reproduire les résultats pour des situations semblables. Ensuite, les résultats du scénario de zombies montrent que le Framework donne des résultats valides pour un scénario complexe. Finalement, le scénario de bactéries démontre que même s'il y a une intervention de la part d'un utilisateur en cours de simulation, le Framework qualifie correctement les types d'interactions présentes dans le scénario.

Contributions

Le Framework qui a été présenté dans ce document touche plusieurs sphères d'activité. En effet, le projet permet de regrouper plusieurs champs d'étude du génie informatique dans un but commun soit : faciliter l'analyse de l'information fournie par une simulation visuelle interactive. Pour ce faire, le Framework offre des fonctionnalités de détection et d'identification d'interactions automatique entre les entités du système complexe simulé. Le projet touche donc à la simulation visuelle interactive, au biomimétisme, à la modélisation, aux systèmes experts et aux systèmes complexes.

Tout d'abord, il va de soi que la principale contribution de la thèse se situe au niveau de la simulation visuelle interactive. En effet, la raison d'être du Framework est d'assister l'utilisateur dans sa compréhension d'un système complexe en l'informant sur le type d'interactions présentes dans le scénario. Pour ce faire, l'architecture logicielle et les algorithmes proposés offrent une nouvelle approche d'aide à la décision. D'une part, l'architecture du Framework se base sur le domaine de l'écologie, plus précisément sur le paradigme des écosystèmes. Cette représentation d'un scénario permet de simplifier la compréhension du système dans son ensemble. En effet, la structure et le langage utilisés sont reconnus et on été mis à l'épreuve dans une multitude de contextes dans le domaine de l'écologie. D'autre part, l'algorithme de détection d'interactions permet d'identifier et catégoriser les interactions présentes dans le scénario. Le processus de détection et d'identification répertorie les interactions en fonction de leur apport sur les *Organismes* impliqués. De cette façon, il est possible de répertorier les interactions en se basant sur les interactions biologiques. Ces dernières couvrent toutes les possibilités d'interactions. Donc, le Framework offre à l'utilisateur de simulation visuelle interactive de constater rapidement quelles entités interagissent et quel type d'interactions est impliqué pour le scénario simulé en utilisant un vocabulaire concis, simple et familier.

Le domaine des systèmes complexes bénéficie aussi de la nouvelle approche présentée dans ce document. Désormais, une partie de l'analyse du système complexe à l'étude se fait automatiquement en détectant et en identifiant les interactions présentes dans le scénario simulé. Ceci permet de simplifier la tâche de l'utilisateur

en lui enlevant le fardeau de la détection et de l'identification des d'interactions régissant le scénario. Donc, le paradigme proposé permet d'aider les prises de décision en procédant, au préalable, à une phase de calcul.

Ensuite, le domaine du biomimétisme, qui consiste à reproduire artificiellement les propriétés essentielles d'un système biologique, est aussi touché par les travaux de recherche. Effectivement, autant le Framework que les algorithmes l'accompagnant s'inspirent de la nature. D'une part, la structure du Framework reproduit la structure d'un écosystème. D'autre part, l'identification suit la classification des interactions biologiques. De cette façon, il est désormais possible d'identifier toutes les interactions présentes dans un scénario de façon automatique, claire et détaillée.

Le domaine de la modélisation est aussi abordé dans le projet de recherche. En effet, le Framework permet de modéliser des scénarios s'inspirant de la nature et, plus précisément, des écosystèmes. De cette façon, le vocabulaire accompagnant le Framework sert désormais de base de connaissances afin de généraliser les termes utilisés dans la modélisation de systèmes complexes dictés par les interactions entre ses parties.

Finalement, le Framework proposé peut être vu comme un nouveau type de systèmes experts. En effet, le Framework prend la place d'un expert qui informe l'utilisateur sur le type d'interactions régissant le système complexe à l'étude. Donc, il offre une approche novatrice qui permet d'effectuer un raisonnement à partir des propriétés de chaque entité simulée pour ensuite informer l'utilisateur des différents comportements présents dans le système à l'étude.

En résumé, le projet de thèse a un caractère intégrateur. En effet, il touche une multitude de domaine d'études dans le but ultime de simplifier l'analyse de l'information fournie par une simulation visuelle interactive. C'est donc pour cette raison que le projet de thèse contribue à la simulation visuelle interactive, les systèmes complexes, les systèmes experts, le biomimétisme et la modélisation.

Intégration

Le Framework proposé dans le cadre du projet de thèse a comme objectif principal de faciliter l'analyse d'une simulation visuelle interactive par l'ajout de mécanismes d'organisation de l'information pertinente au système simulé et par la détection automatique des interactions présentes dans le scénario. Par contre, le Framework développé n'est pas un logiciel à part entière. En effet, pour les besoin de la thèse, il a été nécessaire d'utiliser l'engin de simulation Netlogo pour générer les données de simulation. De plus, Matlab a été utilisé pour l'affichage des résultats. Donc, à l'état actuel, le Framework repose sur d'autres logiciels afin qu'il puisse offrir

ses fonctionnalités. La Figure 83 présente la façon dont le Framework devrait être intégré à d'autres logiciels existants.

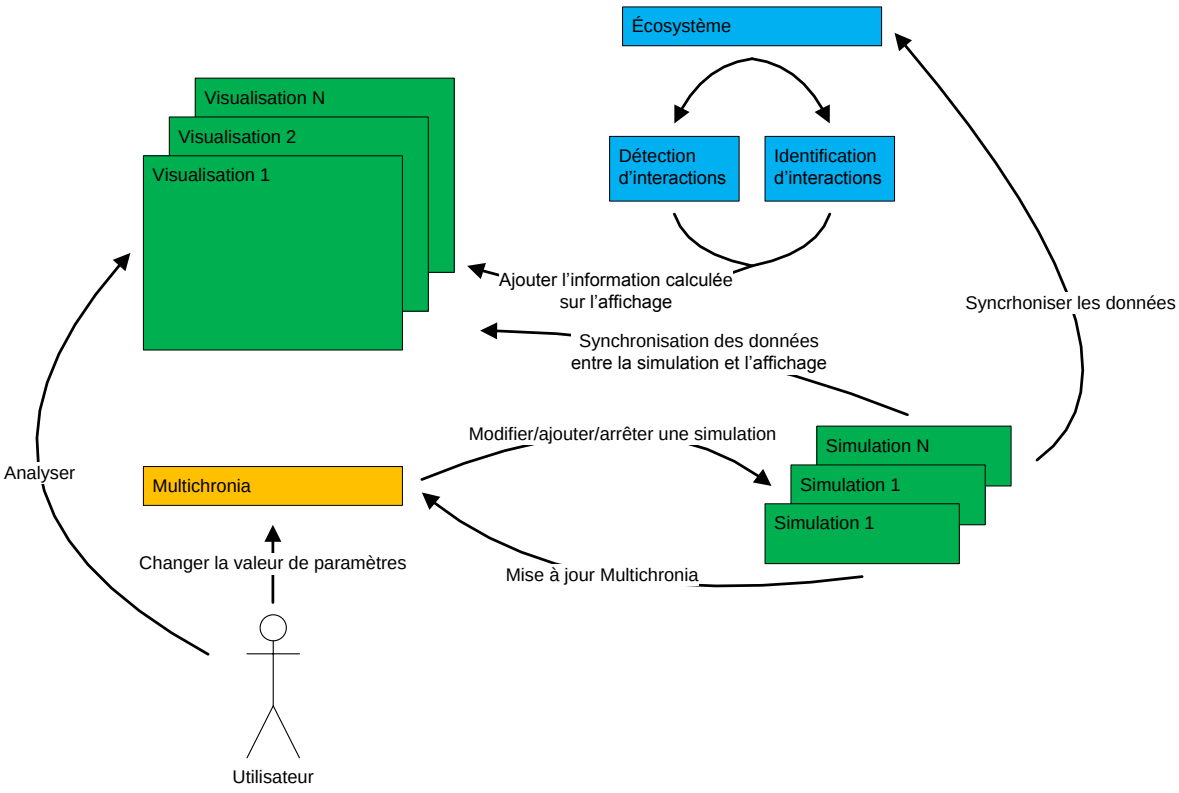


Figure 83 Intégration du Framework à d'autres modules logiciels

D'une part, le Framework doit être intégré à un engin de simulation qui s'occupera du déroulement de la simulation et de la cohérence des données générées (partie verte sur la Figure 83). De plus, l'engin serait utilisé pour afficher de l'information supplémentaire à l'utilisateur. Donc, l'engin fournira l'information nécessaire au Framework afin qu'il puisse analyser la simulation et informer l'utilisateur en superposant les résultats calculés sur l'affichage standard de l'engin de simulation. D'autre part, le Framework peut être jumelé à Multichronia [92] qui permet de lancer en parallèle une même simulation mais avec des paramètres différents. Ceci permet de comparer les résultats d'une même simulation avec des paramètres de configuration d'un scénario différents. Le Framework peut donc aider l'utilisateur dans sa prise de décision en effectuant un traitement préliminaire sur les données de simulation pour ensuite afficher les résultats de ces traitements à l'utilisateur.

La Figure 84 présente l'interface graphique que pourrait implanter une application intégrée.

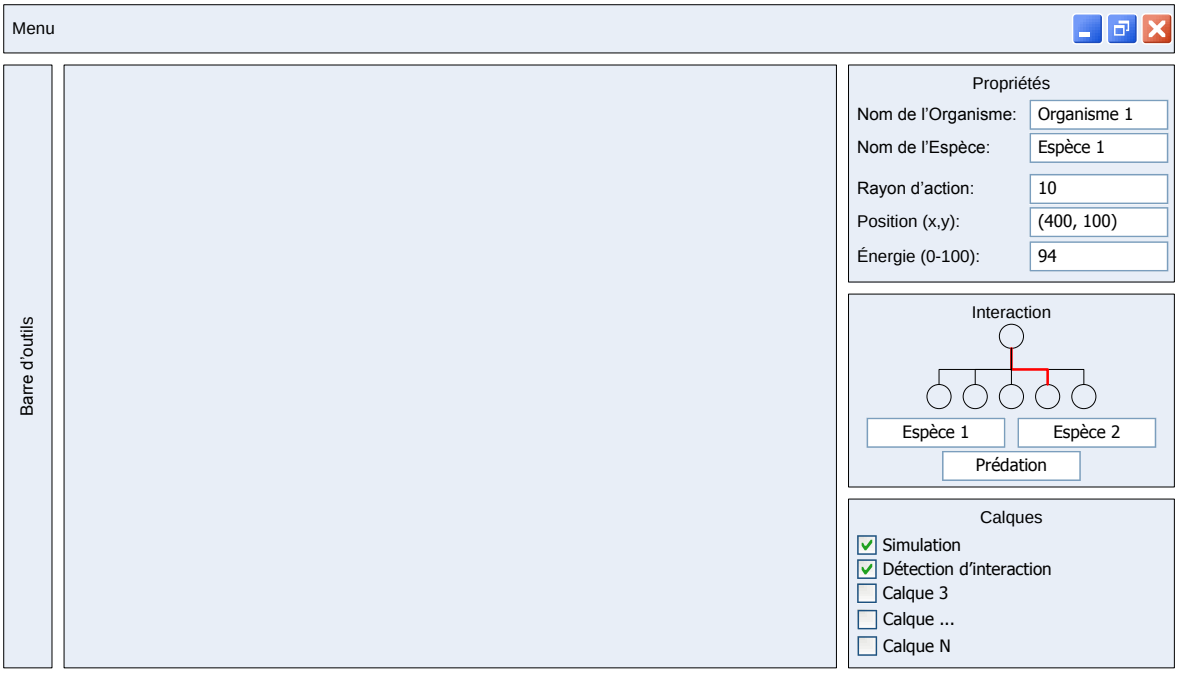


Figure 84 Interface graphique pour un outil de simulation intégré

L'interface s'inspire de celui du logiciel Photoshop du fabriquant Adobe. Elle est composée de trois composantes. La première regroupe les barres de menu et d'outils. Elles offrent toutes deux des fonctionnalités d'édition de simulations (incluant Multichronia), d'exécution de simulations et de contrôle de contrôle standard du logiciel. La seconde composante renferme les éléments spécifiques à la simulation en cours. C'est à cet endroit qu'est affichée l'information sur les *Organismes*, les *Espèces* et les *Interactions*. De plus, cette section renferme le contrôle des calques qui peuvent être superposées sur l'affichage graphique de la simulation qui se trouve à être la dernière composante du logiciel.

Travaux futurs

Les résultats démontrent que l'approche proposée et implantée dans le Framework permet de mieux comprendre l'évolution de la simulation en s'attardant sur les interactions présentes dans le scénario. Par contre, quelques éléments pourraient être ajoutés ou modifiés afin d'améliorer l'information fournie à l'utilisateur et, par le fait même, rendre le Framework encore plus malléable. Tout d'abord, le calcul de l'*Habitat* n'offre pas la possibilité d'être de forme concave et divisé en plusieurs sous-*Habitats*. Ce qui implique que dans l'état actuel des choses, peu importe la distance séparant deux *Individus* d'un même *Espèce,* l'*Habitat*

calculé sera toujours fermé et convexe. Il est à noter que l'introduction d'*Habitat* troué et/ou concave implique la nécessité de revoir l'algorithme calculant l'intersection entre deux *Habitats*. Par contre, de par la nature versatile du Framework, il serait aisé de modifier les algorithmes en question au besoin si la nécessité au besoin. Ensuite, le *RayonAction* pourrait être modifié afin d'y ajouter une incertitude lors de la détection d'autres *Individus* dans les entourages. En effet, le *RayonAction* pourrait utiliser une atténuation de façon à ce que plus un *Individu* est éloigné, plus il est difficile de le détecter. Finalement, coupler le Framework à un engin de simulation pourrait s'avérer essentiel afin qu'il soit plus aisé d'utiliser ses fonctionnalités. À l'état actuel, plusieurs étapes sont nécessaires afin de pouvoir obtenir les résultats. Donc, un environnement complet pour la simulation visuelle interactive assisté incluant des outils de modélisation et d'analyse automatique serait vraisemblablement bénéfique.

Bibliographie

[1] Jerry Banks, John S. Carson II, Barry L. Nelson, and David M. Nicol, *Discrete-Event System Simulation Forth Edition*. New Jersey: Pearson Prentice Hall, 2005.

[2] Henry Markram, "The Blue Brain Project," *Nature Reviews Neuroscience 7*, pp. 153-160, Février 2006. [Online]. http://bluebrain.epfl.ch/

[3] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Interaction

[4] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/écosystème

[5] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Cohérence

[6] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Interaction_biologique

[7] Paul A. Fishwick,. Fishwick.

[8] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Design_of_experiments

[9] William R. Myers, *Encyclopedia of Biopharmaceutical Statistics, Second Edition*. London: Informa Healthcare, 2003.

[10] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Verification_and_Validation_(software)

[11] Rational Software Corporation, "Rational Unified Process: Best Practices For Software Development Teams,".

[12] Philippe Kruchten, "A Rational Development Process," *CrossTalk*, pp. 11-16, 1996.

[13] Robert E. Shannon, "Introduction To The Art And Science Of Simulation," , Washinton, DC, 1998, pp. 7-14.

[14] Jerry Banks and Randall Gibson, "Don't Simulate When," Septembre 1997.

[15] Frederick S. Hillier and Gerald J. Lieberman, *Introduction To Operations Research, Eighth Edition*. New-York: McGraw-Hill, 2005.

[16] C. Dennis Pegden, Randall P. Sadowski, and Robert E. Shannon, *Introduction to Simulation Using SIMAN, 2nd edition*. New-York: McGraw-Hill, 1995.

[17] Winter Simulation Conference (WSC). [Online]. http://www.wintersim.org/

[18] American Statistical Association (ASA). [Online]. http://www.amstat.org/

[19] Association for Computing Machinery: Special Interest Group on Simulation (ACM/SIGSIM).

[Online]. http://www.sigsim.org/

[20] Institute of Electrical and Electronics Engineers: Systems, Man, and Cybernetics Society (IEEE/SMC). [Online]. http://www.ieee-smc.org/

[21] Institute for Operations Research and the Management Sciences: Simulation Society (INFORMS-SIM). [Online]. http://www.informs-sim.org/

[22] Institute ofInductrial Engineers (IIE). [Online]. http://www.iienet2.org/Default.aspx

[23] National Institute of Standards and Technology (NIST). [Online]. http://www.nist.gov/index.html

[24] The Society for Modeling and Simulation International (SCS). [Online]. http://www.scs.org/

[25] Winter Simulation Conference (WSC). [Online]. http://www.wintersim.org/prog09wsc.htm

[26] Robert M. O'Keefe and Peter C. Bell, "Visual Interactive Simulation — History, recent developments, and major issues," vol. 49, no. 3, 1987.

[27] Robert D. Hurrion, "An Investigation of Visual Interactive Simulation Methods Using the Job-Shop Scheduling Problem," vol. 29, no. 11, 1978.

[28] DoD: M&S, DoD Modeling and Simulaiton (M&S) Glossary, January 1998.

[29] K. J. MUSSELMAN, "TESS as a Catalyst for Change," , Los Angeles, CA, 1986.

[30] Bernanrd P. Zeigler, Herbert Praehofer, and Tag Gon Kim, *Theory of Modeling and Simulation Second Edition: Integrating Discrete Event and Continous Complex Dynamic Systems*. San-Diego, CA: Academic Press, 2000.

[31] Bernard Pavard and Julie Dugdale. An Introduction to Complexity in Social Science. [Online]. http://www.irit.fr/COSI/training/complexity-tutorial/complexity-tutorial.htm

[32] Cliff Joslyn and Luis M. Rocha, "Towards Semiotic Agent-Based Models of Socio-Technical Organizations," , Tucson, Arizona, 2000.

[33] Luis M. Rocha. Complex Systems Modeling: Using Metaphors From Nature in Simulation and Scientific Models. [Online]. http://informatics.indiana.edu/rocha/complex/csm.html

[34] Complex Systems Society (CSS). Complex Systems Society. [Online]. http://css.csregistry.org/tiki-index.php

[35] Webopedia. [Online]. http://www.webopedia.com/TERM/N/nondeterministic_system.html

[36] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Radioactive_decay

[37] Gene E. Robinson, "Regulation of Division of Labor in Insect Societies," *Annual Review of Entomology*, vol. 37, pp. 637-665, Janvier 1992.

[38] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Programmation_orientée_objet

[39] François Bernier, Denis Poussart, Denis Laurendeau, and Martin Simoneau-Drolet, "Interaction-Centric Modelling for Interactive Virtual Worlds: the APIA Approach," , Québec, Canada, 2002.

[40] Grady Booch, *Object Oriented Design with Applications*. Redwood City, CA: Benjamin Cummings, 1991.

[41] Morton E. Winston, Roger Chaffin, and Douglas Herrmann, "A Taxonomy of Part-Whole Relations," *Cognitive Science*, vol. 11, no. 4, 1987.

[42] Thorsten Wiegand, Florian Jeltsch, Ilkka Hanski, and Volker Grimm, "Using pattern-oriented modeling for revealing hidden information: a key for reconciling ecological theory and application," vol. 100, no. 2, pp. 209–222, Février 2003.

[43] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Top-down_and_bottom-up_design

[44] Volker Grimm and Steven F. Railsback, *Individual-based Modeling and Ecology*. Princeton, NJ.: Princeton University Press, 2005.

[45] Volker Grimm and al., "Pattern-oriented modelling in population ecology," vol. 186, pp. 151-166, 1996.

[46] James D. Watson, *The Double Helix: A Personal Account of the Discovery of the Structure of DNA*. New York: Atheneum, 1968.

[47] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Medawar_zone

[48] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Agent-based_model

[49] Stuart Russell and Peter Norvig, *Artificial intelligence: A modern approach*. New Jersey: Prentice Hall, 1995.

[50] Steven H. Strogatz, *Nonlinear Dynamics And Chaos: With Applications To Physics, Biology, Chemistry, And Engineering*. Boulder, CO: Westview Press, 2001.

[51] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Rationalité

[52] Lotfi A. Zadeh. Scholarpedia. [Online]. http://www.scholarpedia.org/article/Fuzzy_logic

[53] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Fuzzy_logic

[54] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Degrees_of_truth

[55] Matlab. [Online]. http://www.mathworks.com/help/toolbox/fuzzy/fp351dup8.html

[56] Qiang Luo and Dongyun Yi, "Interaction Detection via Probabilistic Fuzzy Logic," , Jinan Shandong, China, 2008.

[57] A. H. Meghdadi and M.-R. Akbarzadeh-T., "Uncertainty modeling through probabilistic fuzzy systems," , College Park, MD, 2003.

[58] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence

[59] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Design_of_experiments

[60] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Connexionnisme

[61] Marc Parizeau. Algorithmes de l'ingénieur II. [Online]. http://wcours.gel.ulaval.ca/2010/h/IFT3901/default/5notes/Intro-PMC.pdf

[62] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Réseau_de_neurones_artificiel

[63] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Decision_tree

[64] Matlab. [Online]. http://www.mathworks.com/products/statistics/demos.html?file=/products/demos/shipping/stats/classdemo.html

[65] Michel Serres, "Retour au Contrat naturel," Mai 2006.

[66] Neil A. Campbell and Richard Mathieu, *Biologie*. Saint-Laurent, Québec: Édition de Renouveau Pédagogique inc., 1995.

[67] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Ressource_naturelle

[68] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Organisme_(physiologie)

[69] The Free Dictionary. [Online]. http://fr.thefreedictionary.com/propriété

[70] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Habitat_(écologie)

[71] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Tree_(data_structure)

[72] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Architecture_logicielle

[73] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Patron_de_conception

[74] James Rumbaugh, Ivar Jacobson, and Grady Booch, *The Unified Modeling Language Reference Manual*. Indianapolis, IL: Addison-Wesley, 1999.

[75] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Composition_(programmation)

[76] MathWorks. [Online]. http://www.mathworks.com/products/matlab/

[77] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Template_(programming)

[78] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Generics_in_Java

[79] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Weak_typing

[80] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Quantification

[81] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Viabilité

[82] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Liste_rouge_de_l'UICN

[83] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Dynamique_des_populations

[84] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Capacité_porteuse

[85] Wikipedia. [Online]. http://en.wikipedia.org/wiki/Polygon

[86] Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Enveloppe_convexe

[87] MathWorks. [Online]. http://www.mathworks.com/help/techdoc/ref/convhull.html

[88] Cyril Briquet. (2007, Février) Department of EE & CS, University of Liège. [Online]. http://www.montefiore.ulg.ac.be/~briquet/algo3-chull-20070206.pdf

[89] The Center for Connected Learning (CCL) and Computer-Based Modeling. [Online]. http://ccl.northwestern.edu/netlogo/

[90] Joshua R. Nahum, Brittany N. Harding, and Kerr Benjamin, "Evolution of restraint in a structured rock–paper–scissors community," *Proceedings of the National Academy of Sciences*, vol. 08, no. Supplement 2, pp. 10831-10838, juin 2011.

[91] (2011, août) Wikipedia. [Online]. http://fr.wikipedia.org/wiki/Division_cellulaire

[92] François Rioux, *CONCEPTION ET MISE EN OEUVRE DE MULTICHRONIA, UN CADRE CONCEPTUEL DE SIMULATION VISUELLE INTERACTIVE*, Université Laval, Ed. Québec, Canada: Les Presses de l'Université Laval, 2009.

[93] Wikipedia. [Online]. http://fr.wiktionary.org/wiki/intransitif

[94] Le grand dictionnaire terminologique. [Online]. http://www.oqlf.gouv.qc.ca/ressources/gdt.html