Vers une spécification des modèles de simulation de systèmes complexes

Raphaël Duboz, Bruno Bonté, Gauthier Quesnel

Vers une spécification des modèles de simulation de systèmes complexes

Raphaël Duboz [1,2] — Bruno Bonté [1] — Gauthier Quesnel [3]

[1] *CIRAD, UPR AGIRs 22, Campus International de Baillarguet, 34398 Montpellier Cedex 5, France*

raphael.duboz@cirad.fr

[2] *Asian Institute of Technology, Computer Science and Information Managment Laboratory, P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand*

[3] *INRA, UR875 Biométrie et Intelligence Artificielle, F-31326 Castanet-Tolosan, France*

gauthier.quesnel@toulouse.inra.fr

RÉSUMÉ. *L'adoption d'un protocole partagé pour communiquer les modèles de simulation de systèmes complexes de façon rigoureuse est actuellement nécéssaire. Ce constat est soutenu par la demande croissante en modélisation, non seulement pour comprendre la réalité qui nous entoure, mais également pour faire de l'aide à la décision. La spécification rigoureuse des ces modèles de simulation est essentielle car elle permet leur communication, leur reproduction et donc leur vérification. Dans ce papier, nous proposons de réunir les propositions de deux communautés en modélisation et simulation. La première est la communauté des chercheurs en modélisation individus centrés, ou modélisation à base d'agents. Cette communauté propose un protocole, définit par V. Grimm et al. en 2006, revu en 2010. La seconde est la proposition issue de la communauté des chercheurs en simulation au sens large, qui existe autour de la théorie initiée par B. Zeigler en 1976. Nous faisons le lien entre ces deux communautés à partir d'un article de G. Aumann de 2007 qui introduit la théorie de Zeigler en écologie. Ici, nous proposons et décrivons le cadre général d'une méthode de spécification des modèles de simulation qui permet la reproduction et la vérification, et donc augmente la confiance autour des modèles de simulation de systèmes complexes. Nous proposons l'utilisation d'un formalisme systémique pour la spécification de ce protocole. Néanmoins les aspects techniques et purement formels de la spécification, même s'ils sont fondamentaux, ne sont pas abordés dans ce papier car ils ne*

Studia Informatica Universalis., pages 7 à 37

sont pas nécessaires à sa compréhension. Ils devront néanmoins être présentés prochainement dans leur totalité pour que le protocole proposé ici soit effectif.

ABSTRACT. *The adoption of a common and rigorous protocol for the communication of complex systems simulation models is necessary. The increasing demand in modelling and simulation for the understanding of systems and for providing decision making tools imposes to complex models to be replicable and reliable. Therefore, the adoption of a common and rigorous protocol for the communication of complex simulation models of complex systems is necessary. Rigorous model specification of models enables their communication, reproduction and verification. In this paper, we propose a first step towards the unification of two propositions made by two different communities. The first community develops Individual Based Models or Agent Based Models. This community proposes a protocol defined by V. Grimm et al. in 2006, reviewed in 2010. The second proposition comes from the simulation community and is based on the theory of modelling and simulation initiated by B.P. Zeigler in 1976. We link these two propositions using a paper by G. Aumann in 2007 introducing Zeigler's theory in ecology. Here, we propose a general specification framework for the specification of simulation models of complexes systems, which enable reproduction and verification, increasing the reliability of models. We propose the use of a systemic formalism for the specification of models. Nevertheless, even if they are of great importance, formal and technical aspects are not deeply present here as they are not necessary for the understanding of the general purpose of this paper. They shall be provide soon for the present general framework to be effective.*

MOTS-CLÉS : *Système, Formalisme, Communication, Protocole, cadre expérimental, Multiagents, ODD.*

KEYWORDS: *System, Formalism, Communication, Protocol, Experimental Frame, Multiagents, ODD.*

1. Introduction

La communication rigoureuse d'un protocole expérimental est au cœur de l'activité scientifique. Elle permet la reproduction sans ambiguïté de l'expérience, et donc la vérification des résultats par les pairs. Dans les sciences expérimentales, la description des protocoles et des appareils de mesures peut être longue et laborieuse. Elle consiste en une documentation technique qui décrit les appareils, les phénomènes physico-chimiques en jeu, les calculs associés pour restituer l'information etc. Ces appareils font l'objet de procédures de normalisation pour garantir la fiabilité et la reproductibilité des résultats. Il devrait en être de même pour les modèles de simulation, c'est-à-dire les modèles qui nécessitent de passer par une résolution numérique, ou disons plus généralement par un calcul sur ordinateur, pour en explorer le comportement, la dynamique. Actuellement, les simulations sont considérées comme des expériences virtuelles, ou expériences de seconds ordres [Var06]. Elles permettent de résoudre des équations autrement insolubles ou d'exécuter des algorithmes qui simulent le comportement d'un système donné. La simulation prend ainsi une place de plus en plus importante dans l'étude des systèmes complexes, en tant qu'objet de substitution au réel. Ainsi, faire des expériences à l'aide d'un modèle de simulation devient une pratique courante pour tester des hypothèses sur les dynamiques et les structures (i.e. les composants et leurs relations) des systèmes complexes. Néanmoins, les modèles de simulation peuvent être compliqués, c'est-à-dire longs et difficiles à décrire, et leurs comportements aussi complexes que les phénomènes qu'ils sont censés représenter. Ces deux points sont à l'origine de la réticence de certains scientifiques à utiliser ce type de modèle. Ces scientifiques mettent en doute la confiance que l'on peut avoir dans les résultats obtenus par simulation, et donc dans les conclusions que l'on peut en tirer (voir à ce sujet l'introduction de l'article de Graig A. Aumann [Aum07], pour un argumentaire développé pour l'écologie).

Il ne s'agît pas pour nous de discuter ici de l'intérêt d'avoir ou non des modèles de simulation très détaillés. Il est question de faire le point sur la spécification des modèles de simulation en établissant un lien entre deux approches différentes. La première approche est un protocole de communication des modèles individus centrés (IBM pour *Indi-*

vidual Based Model) ou modèles de simulation multiagents (ABS pour *Agents Based Simulation model*) développée par V. Grimm et al. en 2006 [VUF+06], revu et amélioré en 2010 [GUD+10]. La seconde est celle développée depuis les années 70 par B.P. Zeigler et ses évolutions récentes [Zei76, ZKP00, Wai10] dans le cadre de la Théorie de la Modélisation et de la Simulation (TMS).

La problématique de la communication et de la vérification est centrale et bien identifiée, à la fois par la communauté de la TMS ainsi que par la communauté des concepteurs et utilisateurs d'IBM et d'ABS. La première communauté s'intéresse surtout (mais pas seulement) à des simulations orientée vers l'optimisation et la conception de systèmes industriels, ou plus généralement artificiels. Dans ce cas, le système est suffisamment bien connus pour pouvoir en reproduire le comportement. La deuxième communauté utilise principalement les simulations pour évaluer des représentations possibles de systèmes afin d'augmenter la connaissance sur les systèmes en question. Dans les deux cas, une spécification rigoureuse du modèle est nécessaire. Il s'avère que la communauté TMS propose un cadre général qui peut bénéficier à la communauté ABS, qui en retour pose des problèmes nouveaux à la TMS dans le domaine de la complexité [Aum07]. C'est ce qui a motivé cet article.

Dans une première partie, nous ferons un point rapide sur la spécification des ABS en général. Nous rappellerons ensuite certains des éléments du cadre théorique de la modélisation des systèmes. Nous ferons enfin une proposition méthodologique pour la spécification des modèles de simulation de systèmes complexes et nous la discuterons.

2. Spécification des simulations à base d'agents

Le cas des ABS est intéressant car le paradigme de modélisation des Systèmes Multi-Agents (SMA), dont ils sont issus, n'est pas compris, et par conséquent formalisés, de façon univoque (le livre récent de J.P. Treuil et al. illustre bien ce fait par les différents exemples de simulation à base d'agents qu'il présente [TDZ08]). La souplesse des définitions du paradigme agent n'est pas une tare, c'est un avantage qui permet à différentes disciplines aux définitions hétérogènes de se ren-

contrer sur un terrain sémantique commun. Néanmoins, quand vient le moment des choix précis de modélisation, c'est-à-dire des formalismes et des algorithmes à implémenter pour simuler le modèle, les ambiguïtés doivent être levées, au moins entre le modèle et le simulateur, nous y reviendrons plus tard.

Prenons le cas emblématique de NetLogo, un logiciel de modélisation et simulation très utilisé dans le domaine des ABS [Wil99]. NetLogo propose un langage dit « de modélisation ». Ce langage consiste en des primitives de très haut niveau comme « *create-turtles* » pour créer des agents ou « *move-to* » pour déplacer un agent. La syntaxe du langage NetLogo permet d'écrire des algorithmes qui simulent le comportement des agents, des groupes d'agents et de leurs environnements. Au delà des aspects syntaxiques, ce type de langage de programmation agent est conditionné par les hypothèses sur ce qu'est un agent, son environnement, une interaction, une communication etc. Nous ne pouvons donc pas l'utiliser pour reproduire et vérifier des modèles de simulation à base d'agents conçu pour un autre logiciel, qui proposera une implémentation différente, même légèrement, des ABS. Il nous faudra ré-implémenter complètement le modèle, mais « *à l'aveugle* », car les hypothèses de conception ne sont pas clairement formulées. Des tentatives de mise au point de langages de modélisation agent existent. Par exemple en utilisant UML et en générant le code [PTW07], ou les langages métiers [Hah08], ou même des formalisation systémique [DRQ04], mais il semble qu'ils soient très peu utilisés de par leur manque d'expressivité et surtout de généricité [SDS10]. Pourtant chaque simulation d'agents n'est pas seulement un cas particulier, et un protocole à été proposé pour leur spécification.

2.1. *Ambiguïtés du protocole «* Overview Desing concepts Details *» (ODD)*

Une méthodologie de description des ABS à été proposée par V. Grimm et al. en 2006 [VUF+06], revue et améliorée en 2010 [GUD+10], il s'agît du protocole ODD pour « *Overview Desing concepts Details* ». Ce protocole améliore la communication des ABS, mais des imprécisions et des ambiguïtés demeurent qui empêchent, à

notre sens, l'utilisation d'ODD pour la communication et la vérification des modèles de simulation, ce qui était pourtant l'objectif affiché par ce protocole dès 2006, lorsque les auteurs disaient qu'un des objectifs à long terme d'ODD est d'arriver à un niveau de description suffisamment précis pour que la description fournie puisse être « convertie directement en un simulateur sans possibilité d'erreurs de programmation ». Les auteurs ajoutent qu'une fois cet exécutable généré, un environnement d'expérimentation de modèle doit être mis à disposition, qui permette de générer, d'observer, d'interpréter, etc. des expériences. Ce dernier point implique que cet environnement d'expérimentation soit lui-même décrit sans ambiguïté. Néanmoins, l'objectif de 2006 ne semble toujours pas atteint, et nous donnons ici des exemples d'ambiguïtés relevées dans le protocole ODD (se référer à l'article [GUD+10] pour les détails du protocole).

2.2. *Confusion entre expérience et modèle*

La différence entre expérience et modèle est bien sûr fondamentale, et nous le discuterons plus loin. Quoi qu'il en soit, les valeurs des paramètres qui font partie de l'expérience faite sur un modèle, ainsi que leur définition, doivent à notre sens apparaître à un seul endroit pour éviter les confusions. Hors dans le protocole ODD, les définitions et les valeurs des paramètres peuvent apparaître à plusieurs endroits : section 7 *Submodels* , section 5 *Initialization* ou dans la définition des valeurs initiales des variables d'états. Ces variables d'états (également appelées attributs dans le protocole ODD) peuvent rester constantes lors de l'exécution du modèle. Si c'est le choix du modélisateur et non une conséquence du calcul, alors une variable qui reste constante n'est plus une variable, par définition, c'est un paramètre. En plus de possibilités multiples de déclaration des paramètres, la distinction n'est pas faite clairement entre variables et paramètres dans le protocole ODD.

2.3. *Définition de la notion de variable d'état*

Le fait de définir une variable d'état comme ne pouvant pas être calculée à partir d'autres variables d'états est également très restrictif. Si

l'on prend l'exemple donné pour décrire le protocole, la distance entre un agent localisé par x et y sur une grille et un service quelconque localisé sur une autre case de la même grille, ne devrait pas être une variable d'état car elle est calculable à partir d'autres variables d'états du modèle. Donc, si la position de l'agent est calculée en fonction de sa vitesse et de son comportement, alors elle non plus n'est pas une variable d'état, pour les même raisons. Nous pouvons alors nous poser la question légitime de savoir qu'est-ce qui est « variable d'état » dans un modèle où presque toutes les valeurs dynamiques sont calculées en fonction d'autres valeurs dynamiques. De plus, cette définition interdit de considérer une valeur agrégée (la somme des agents présents à un instant t par exemple) comme une variable d'état, alors que cette valeur pourrait très bien servir à un agent collectif pour entreprendre une action par exemple. Cette valeur qualifierait bien l'état du collectif à un instant t mais ne serait pas une variable d'état décrite par le protocole ODD.

2.4. *Spécification des modèles*

Dans le protocole ODD, la description des « *sub-models* » peut prendre des formes verbales, mathématiques ou algorithmiques par l'ajout du code informatique complet en annexe du papier. Encore une fois, cette totale liberté d'expression de l'ensemble des modèles composants un ABS ne peut pas être considérée comme une avancée vers une meilleure communication des modèles, car aucune standardisation ou normalisation n'est possible ici du fait de la multiplicité des langages et des simulateurs, qui sont de plus amenés à évoluer, ce qui rend ce type de spécification non pérenne.

Nous pensons qu'il y a parfois une confusion entre modélisation objet et modélisation des systèmes dynamiques dans le protocole ODD (héritage, versus niveau de spécification et base de temps, voir section 3.1). Dans la version révisée du protocole [GUD+10], l'excessive proximité entre la programmation objet et ODD a été identifiée. Les auteurs reviennent notamment sur leur conseil initial d'utiliser les diagrammes de classes UML pour décrire graphiquement la structure des modèles. Ils persistent cependant à décrire les entités de leurs modèles

comme un ensemble d'attributs et de méthodes ou de fonctions, c'est-à-dire comme une classe ou son instance en langage objet. Les auteurs expliquent que le lien entre les entités et les processus associés doivent être décrits dans la partie « ordonnancement » de la spécification du modèle global. Néanmoins, la représentation du temps, et donc de la dynamique des interactions entre agents ou sous-modèles, reste problématique dans cette proposition.

3. Spécification dans la théorie de la modélisation et de la simulation

3.1. *Les entités de base*

Nous rappelons ici les définitions qui sont utilisées dans la suite de cet article. Nous ne donnerons pas ici les définitions formelles, elles sont accessibles avec les références citées dans cette partie. Ces définitions ne sont pas indispensables pour une présentation générale.

Une notion importante est celle de « trajectoire d'états » pour les systèmes dynamiques. Les simulations génèrent des trajectoires d'états, c'est-à-dire des suites de valeurs ou de symboles indexées par le temps. Elles sont la trace tangible du comportement du modèle. Ce sont ces trajectoires d'états qui servent de support aux discussions autour de la vérification du modèle, de sa validation, et de son adéquation avec les questions posées.

Une notion également importante et celle de « niveau de spécification » [Kli85]). Cette notion met en relation le niveau de connaissance que l'on a d'un système donné avec sa spécification formelle. Plus le niveau est bas, moins le système est connu dans ces détails. Par exemple au premier niveau, le niveau 0, on dispose juste de données en entrée et en sortie du système, indexées par le temps, c'est le niveau de l'observation du comportement du système.

Les quatre entités de base en Théorie de la Modélisation et de la Simulation (TMS) sont [ZKP00] :

– *Le système source* : c'est le système étudié, ou plus généralement le système d'intérêt. Il peut être réel ou virtuel. Il est vu comme une

source ou une base de données, de trajectoires d'états. Il est spécifié au niveau 0.

– *Le modèle* : à un niveau de spécification donnée au sens de [Kli85], un modèle est un ensemble de règles, d'équations, de contraintes qui spécifie des relations dynamiques entre des entrées et des sorties. Il s'agît de la description d'un mécanisme de génération de trajectoires de sortie en fonction de trajectoires d'entrée. Cette définition à l'avantage d'avoir une base mathématique dans la théorie des systèmes qui permet la communication non ambiguë des modèles. Elle permet également de considérer un modèle à l'intérieur d'un modèle plus grand, en utilisant des mécanismes de composition par couplage de modèles [ZKP00].

– *Le simulateur* : c'est l'entité qui génère le comportement d'entrée-sortie décrit par le modèle. Il peut s'agir d'un calculateur, d'un mécanisme physique quelconque ou encore du cerveau humain.

– *Le cadre expérimental*.

Cette dernière entité est importante pour la spécification des modèles de simulation car elle englobe le modèle, le simulateur et le système source dans un cadre d'usage (figure 1). Le cadre expérimental permet de mettre en relation le système source et le modèle. Nous le détaillons dans ce qui suit.

3.2. *Spécification des cadres expérimentaux*

A partir des objectifs de modélisation, il est possible de définir un cadre expérimental. C'est précisément cette transformation des objectifs en cadre expérimental qui guide la conception du modèle et les expériences effectuées. Le cadre expérimental définit des observables (des trajectoires d'états observées) pour le systèmes source et le simulateur, qui devront être mis en relation pour statuer sur la validité du modèle, au regard des objectifs de modélisation. Les objectifs peuvent être de deux catégories différentes. Soit il s'agît de simuler un système existant ou virtuel dont on connaît le fonctionnement et la dynamique et dont on veut optimiser certaines parties, tester des configuration différentes, améliorer le fonctionnement etc. Les objectifs sont alors ceux de l'ingénierie des systèmes. Soit il s'agît, comme c'est le cas dans l'étude des systèmes complexes, de simuler un comportement pour inférer des

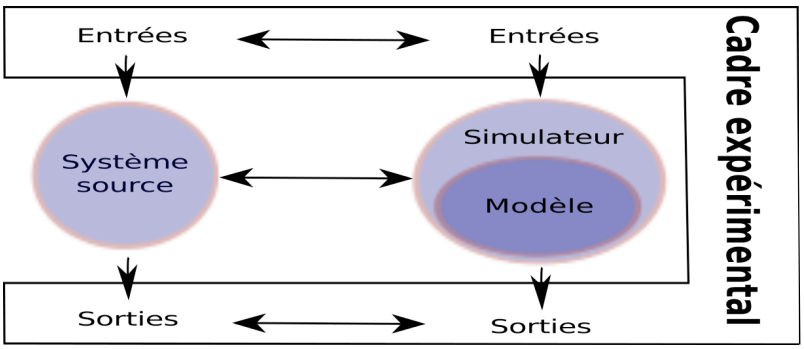

Figure 1 – Cadre général de la modélisation et de la simulation selon Zeigler [ZKP00] et redéssiné par [Aum07]. Les flèches horizontales représentent les relations d'équivalence et d'approximation qui existent entre les entités du cadre. Les flèches verticales représentent un flux de données. La superposition du simulateur et du modèle illustre une relation d'équivalence particulière, un morphisme, que nous discutons dans le texte.

règles de fonctionnement, découvrir, comprendre un mécanisme, le rôle des interactions entre composants du système etc. C'est cette deuxième catégorie d'objectifs qui nous intéresse ici. Nous parlerons dans ce cas plutôt de « question posée » que d'« objectif ».

Pour relier le système source au modèle, la TMS associe la question posée à un cadre expérimental, c'est-à-dire à un contexte dans lequel le comportement du système source nous intéresse. Ce cadre expérimental peut être décrit formellement à différents degrés de spécification [Kli85, ZKP00]. Au niveau le moins précis, le cadre définit uniquement les domaines de définitions des entrés et des sorties du système source (et donc du modèle associé). Au niveau le plus précis, il s'agit d'un ensemble de composants qui interagissent avec le modèle et qui nous permettent à la fois de simuler le contexte d'intérêt, mais aussi d'observer et analyser les trajectoires des variables d'états qui nous intéressent. De manière générale, le cadre expérimental doit décrire rigoureusement l'ensemble des expériences possibles, qui peuvent être réalisées sur le modèle dans le cadre de la question posée. Une telle

approche permet de relier rigoureusement la ou les questions posées, à l'ensemble des expériences. Traore et Muzy proposent une description formelle de cadres expérimentaux à différents niveaux de spécifications [TM06]. La formalisation de ces cadres nous permet d'appliquer un même cadre expérimental à plusieurs modèles, et réciproquement de tester un même modèle dans des cadres expérimentaux différents. Le premier cas nous permet par exemple de trouver le meilleur modèle ou un agencement préférentiel pour notre système si notre objectif est de concevoir un nouveau système. Le second cas nous permet quant-à lui de réutiliser un même modèle pour une question légèrement différente qui considérera le système étudié dans un contexte différent ou sous un point de vue différent. Dans leurs travaux, Traore et Muzy [TM06] partent d'un objectif précis pour arriver à un cadre expérimental entièrement spécifié dans lequel ils peuvent tester des modèles différents pour choisir le plus adapté. Dans leur exemple, ils cherchent un modèle de feu de forêt qui leur permette de prendre des décisions de lutte contre le feu en temps réel. Le cadre expérimental qu'ils définissent, simulé en même temps que le modèle, leur permet de tester différents scénarios d'occurrence des départs de feux tout en contrôlant les deux critères qui les intéressent : le temps d'exécution des simulations et l'erreur entre les résultats de simulation du modèle testé et ceux d'un modèle de référence. Le choix d'un modèle ou d'un autre ne va pas se faire au seul regard de sa faculté à imiter le système, mais sur un compromis entre les deux critères définis dans le cadre expérimental.

Spécifier une expérience consiste à définir un état initial et un ensemble de valeurs des paramètres du cadre expérimental. Par exemple, dans le cas de modèles de feux de forêts, la série temporelle des départs de feux est un paramètre du cadre expérimental. Il s'agit bien de la spécification d'une expérience qui aurait pu être réalisée sur le système. L'expérience consiste à déclencher des feux à certains moments à certains endroits. Le système correspond à la forêt. Les notions de cadre expérimental et d'expérience ainsi définies permettent de distinguer les paramètres de l'expérience de ceux propres au modèle.

Il est important de faire la différence entre un plan d'expériences et un cadre expérimental. Un plan d'expériences spécifie l'ensemble des valeurs prises par les paramètres et l'état initial des variables pour

chaque simulation, ceci à la fois pour le modèle et pour le cadre expérimental. Il va consister en une liste d'instances « cadre expérimental/Modèle », à tester. Un plan d'expérience est définit pour faire une analyse de sensibilité par exemple, ou optimiser un modèle au regard de certains critères définis a priori. Comme le montre le figure 1, le cadre expérimental englobe plus de choses que la spécification des valeurs de paramètres et de variables. Le plan d'expérience contient en fait une réalisation des trajectoires d'états en entrée du modèle, le cadre expérimental définit quand à lui l'ensemble des trajectoires d'états admissibles en entrée.

En TMS, la distinction claire, explicite, du modèle et de son contexte d'utilisation trouve son origine dans les sciences de l'ingénieur et leurs applications. Les objectifs de cette communauté appartiennent à la première catégorie citée plus haut (tester ou évaluer des protocoles ou des agencements de systèmes devant être conçu ou utilisés par l'industrie). Les ingénieurs peuvent donc décrire clairement les attentes qu'ils ont sur leurs systèmes, tout comme ils peuvent limiter leur domaine d'intérêt à certaines conditions d'utilisation. De plus, l'activité de modélisation et simulation en sciences de l'ingénierie (écriture du simulateur incluse) est vue comme la production d'un bien (le modèle et son simulateur) qui doit être réutilisable et qui doit donc être livrée avec ses spécifications, c'est-à-dire avec son cadre expérimental général couvrant l'ensemble des cadres expérimentaux plus restreints pouvant être dérivés selon la question posée par l'utilisateur du modèle.

3.3. *Distinction entre modèle et simulateur*

La distinction entre modèle et simulateur est centrale en TMS. Selon la définition donnée section 3.1, un modèle correspond à une spécification. La transformation de cette spécification en une suite d'instructions exécutables par un ordinateur correspond à l'implémentation du modèle. Selon la définition, le code informatique d'exécution d'un modèle peut être considéré comme équivalent à sa spécification. Néanmoins, l'information présente dans ce code est spécifique à un langage et bien plus détaillée que l'information nécessaire et suffisante à la communication du modèle. Dans les faits, les modèles sont donc spécifiés

dans des formalismes, mathématiques ou non, qui contiennent en théorie l'ensemble de l'information nécessaire pour les décrire. L'étude de la relation d'équivalence entre le modèle est le simulateur est appelée « vérification » en TMS. Elle est, nous l'aurons compris, différente de la notion de « validation » car elle n'étudie pas la relation avec le système source. Cette activité de vérification consiste donc à s'assurer que le simulateur génère bien le comportement spécifié par le modèle. En effet, ce comportement dépend du langage de programmation, du compilateur ou même de l'architecture matérielle (par exemple, en fonction des optimisations du calculs des nombres flottants, des compilateurs, des systèmes d'exploitation, on aura pas exactement les mêmes résultats sur des architectures 32 ou 64 bits). Il dépend aussi de la phase d'implémentation et de ses erreurs potentielles. La vérification formelle du comportement des modèles de systèmes dynamiques est impossible dans beaucoup de cas, notamment lorsque le comportement est stochastique et que l'ensemble des états atteignables par le modèle est infini. Néanmoins les communautés de la TMS et du génie logiciel propose des solutions dans ce domaine de recherche toujours d'actualité [BCK09, LW05].

La vérification repose sur le concept d'homomorphisme qui définit si des systèmes sont équivalents en terme de transitions d'états [ZKP00]. Ce concept s'applique de façon général entre des systèmes quelconques. Prenons par exemple deux systèmes S et S'. Le systèmes S peut être le système source et S' un système qui simule de façon simplifiée le comportement de S. Nous pourrons dire que S et S' sont homomorphes si pour toute transition d'états dans S, il existe une seule ou une suite de transitions d'états dans S' qui commence et finit par les états correspondants dans S. Si le système S est un modèle, S' les algorithmes du simulateur correspondant au modèle et S'' un calculateur qui exécute les instructions et que nous considérons un changement de l'état A vers l'état B, nous avons pour les trois systèmes considérés :

1) modèle
$S : A \rightarrow B$

2) algorithmes
$S' : A' \rightarrow A'_1 \rightarrow A'_2 \rightarrow \ldots \rightarrow A'_m \rightarrow B'$

3) simulateur
$S'' : A'' \rightarrow A''_1 \rightarrow A''_{11} \rightarrow A''_{12} \rightarrow \ldots A''_{1n} \rightarrow A''_2 \rightarrow ... \rightarrow A''_m \rightarrow B''$

Ou, pour paraphraser les trois points précédents, il existe un ensemble de micro-transitions d'état dans un simulateur permettant de générer la transition décrite au niveau du modèle. Nous dirons qu'un simulateur est vérifié s'il y a un homomorphisme entre S et S". Nous voyons bien sur cet exemple que cela implique également un homomorphisme avec le système définit par les algorithmes de simulation. L'homomorphisme entre S' et S" est assuré par le compilateur ou l'interpréteur du code informatique correspondant aux algorithmes de simulation. Même si des erreurs peuvent exister à ce niveau elle sont principalement d'ordre matérielles et nous n'en parlerons pas ici. La correspondance entre S et S' est plus compliquée qu'entre S' et S". Si cette correspondance est calculable, alors cela revient au cas de la correspondance entre S' et S". C'est le cas lorsque l'on dispose d'un langage de modélisation directement compilable ou interprétable (il en existe de très nombreux développés pour des formalismes particuliers comme les *states charts* d'UML, les réseaux de Petri, les automates à états etc). Néanmoins, ces langages imposent un type de modélisation car ils sont basés sur un formalisme particulier. En TMS, la distinction du modèle et du simulateur à permis de développer des outils qui permettent l'intégration de modèle hétérogènes [DRQ04], c'est-à-dire des compositions de modèles pluriformalisés augmentant le potentiel de description d'un système source par un modèle. C'est ce type de multimodèles qui sont difficiles à spécifier en raison de l'articulation compliquée des modèles composants. Là encore, la TMS propose une solution intéressante.

3.4. *Propriété de fermeture*

En TMS, le couplage de plusieurs modèles est vu comme l'union des états et des fonctions des modèles couplés. Le résultat de cette union est un nouveau modèle qui a un comportement équivalent au modèle couplé dont il est issu. L'équivalence du modèle atomique et du modèle couplé est appelée propriété de fermeture sous couplage [ZKP00]. Cette propriété peut être démontrée dans un formalisme donné, et a des implications majeures en terme de modélisation. Par exemple, les en-

trées/sorties d'un modèle couplé et d'un modèle élémentaire sont spécifiées de la même manière i.e. les couplages en entrée et sortie du modèle sont formalisés indépendamment de la structure interne des modèles et reste identique quelque soit le niveau de couplage. Ceci permet de définir des interfaces génériques pour le couplage de modèles, et donc de permettre la composition et la décomposition des modèles sans réécriture.

La propriété de fermeture garantit également que le choix de la structure du couplage n'ait pas d'influence sur le comportement du modèle, i.e. sur les trajectoires d'états générées. Ceci est très important, car le choix de la granularité en terme de modèles élémentaires n'est pas unique. Cette absence d'influence de la structure de composition du modèle sur les trajectoires des états relègue toute la responsabilité des trajectoires d'états observées aux niveau des choix de modélisation.

3.5. *Théorie des hiérarchies*

En TMS, la possibilité de composition hiérarchique de modèles fait naturellement penser à la théorie des hiérarchies [Bra73]. Dans cette théorie, une entité à un niveau hiérarchique donné (ou niveau d'organisation) contient en substance (physiquement, structurellement) les entités de hiérarchies inférieure et fait partie d'un niveau hiérarchique supérieur [Fei54]. En se basant sue cette théorie, G. Aumann identifie au minimum trois niveaux hiérarchiques pour tout modèle. Il les appelle les niveaux focaux [Aum07]. Il définit le niveau focal N comme le niveau d'intérêt dans le modèle, par exemple le niveau individuel dans un ABS ou le niveau population dans un modèle plus agrégé. C'est le niveau où sont modélisés les mécanismes d'intérêts, les interactions entre variables et composants du modèle. Le niveau focal $N-1$ englobe les composants du modèle qui reproduisent un comportement existant à un niveau d'organisation inférieur. C'est le niveau de fermeture inférieur du modèle. Le niveau focal $N+1$ correspond au comportement du système modélisé dans son ensemble.

Il est important de remarquer que les hiérarchies ainsi définies ne correspondent pas aux niveaux de spécification au sens de G. Klir [Kli85]. Il est par exemple possible d'avoir une très grande connaissance (niveau

de spécification élevé) d'un mécanisme interne à un individu, au niveau focal $N-1$. G. Aumann propose d'associer trois cadres expérimentaux aux trois niveaux focaux. Ce faisant, les trajectoires d'états en entrée et sortie du modèle sont également associées aux trois niveaux focaux du modèle. De notre point de vue, G. Aumann clarifie ainsi l'étude de la dynamique des modèles de simulation de systèmes complexes en mettant en relation la théorie des hiérarchies et les modèles de simulation. Dans ce cadre, nous pouvons distinguer les trajectoires d'états qui sont le résultats de calculs qui reproduisent le comportement d'un sous-système, sans pouvoir explicatif au niveau $N-1$ par exemple, des trajectoires d'états qui peuvent être qualifiées d'émergentes, car résultant des interactions des composants au niveau N et observables au niveau $N+1$.

4. Proposition

4.1. *Présentation générale*

En adoptant une vision systémique des modèles de simulation, nous défendons une approche de construction et de spécification par composition de modèles. Chaque modèle étant conçu dans un cadre expérimental donné (pour répondre à des objectifs), la composition des cadres expérimentaux doit accompagner la composition des modèles [Aum07]. La figure 2 illustre les quatres bases de notre proposition[1]. Il s'agît de séparer les spécifications des modèles de celles des cadres expérimentaux, de celles des structures (qui correspond à la composition des modèles et donc des cadres associés). La base des expériences contient les plans d'expériences, instance des cadres expérimentaux. La structure d'un modèle composé est vue comme un arbre où chaque feuille correspond à un modèle indécomposable et chaque nœud à une composition. A chaque feuille et chaque nœud correspond un cadre expérimental.

L'avantage d'une telle représentation est qu'à partir d'une base de modèles donnée, toute nouvelle composition ne nécessite que la définition d'un nouveau cadre expérimental qui se trouve être la composition des cadres associés aux modèles composants. Les ABS sont un archétype de modèle de simulation de systèmes complexes. Ils concentrent

1. Le mot « base » ici à le même sens que dans « base de données ».

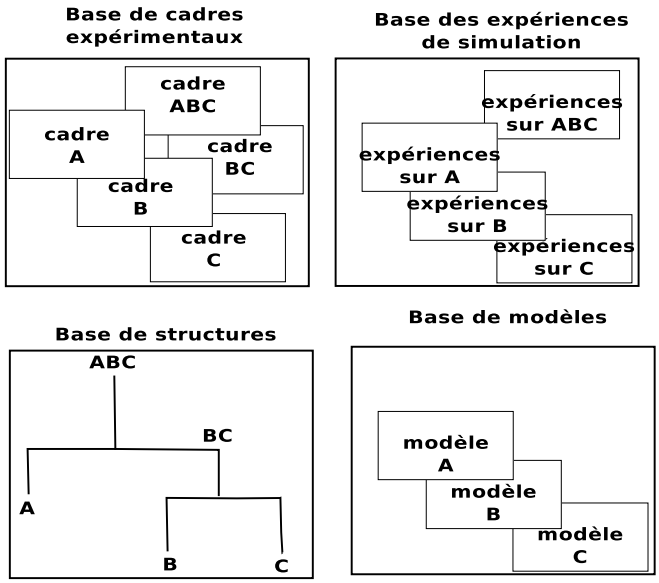

Figure 2 – Les quatre bases dans notre proposition (inspirées de [ZKP00] et [Aum07]). Ce découpage conserve une vision de composition hiérarchique des modèles et des cadre expérimentaux.

l'ensemble des difficultés propres aux modèles très détaillés. Les ABS peuvent être considérés comme des modèles de simulation composés et pluri-formalisés ([DRQ04], [TDZ08]). V. Grimm et al. reconnaissent eux aussi implicitement que les modèles de type ABS sont des compositions de modèles élémentaires puisqu'ils définissent un élément *« sub-models »* qui décrit l'ensemble des composants de l'ABS dans le protocole ODD [GUD+10].

La méthode de communication et de spécification que nous proposons comprend cinq parties qui doivent être considérées dans l'ordre. La spécification complète d'un modèle composé se fait par réitération des cinq parties suivantes, avec possibilité d'omettre une partie si elle n'est pas nécessaire au niveau focal considéré :

1) Communication des objectifs de modélisation et du modèle.
2) Spécification formelle du cadre expérimental du modèle.
3) Spécification formelle du modèle.
4) Spécification formelle du simulateur et du coordinateur associé.
5) Communication des expériences de simulation.

Nous détaillons ces cinq parties dans ce qui suit.

4.2. *Communication des objectifs de modélisation et du modèle*

Cette partie est textuelle. Elle permet à toute personne qui veut utiliser un modèle de comprendre pour quels objectifs le modèle a été construit, quel est son contexte d'utilisation et quelles sont les hypothèses sur le système source qui ont guider les choix de conception. Nous utilisons le terme modèle dans ce qui suit pour désigner aussi bien un modèle composé qu'un modèle élémentaire. La fermeture sous couplage qu'exige notre méthode nous garantit qu'elle s'applique aux deux types de modèle.

4.2.1. *Description générale des objectifs et du modèle*

Pour structurer la description des objectifs, nous considérons tout d'abord la première partie (« *Overview* ») du protocole ODD [GUD+10]. Nous reprenons l'élément « *Purpose* » qui résume

comment le modèle peut répondre à la question de recherche posée. L'élément *« Entities, state variables and scales »* devient « *Entities and scales* ». Il s'agît de décrire les entités du systèmes, leur résolution spatiale et temporelle (à quelles échelles de temps et d'espace les entités sont-elles considérées). Cette partie doit également définir les niveaux focaux (hiérarchiques) du modèle en précisant à quel(s) niveau(x) les objectifs de modélisation (la question posée) seront étudiés. L'élément « *Process overview and scheduling* », est également modifié pour devenir « *Process overview* ». Il ne s'agit pas de décrire les algorithmes d'ordonnancement, ceux-ci devront apparaître dans la partie « spécification du simulateur » de notre méthode. Par contre il est toujours question de décrire les processus qui affectent les entités et de préciser si le modèle est en temps continu, discret ou hybride.

4.2.2. *Communication des hypothèses de construction du modèle*

Nous considérons ici la partie « *Design concepts* » de la dernière version du protocole ODD [GUD+10]. Dans cette partie, il y a originalement onze éléments à renseigner, nous considérons les dix premiers, à savoir « *basic principles, emergence, adaptation, objectives, learning, prediction, sensing, interaction, stochasticity and collectives* ». L'élément « *observation* » est inclut dans la section suivante. Comme il est dit dans le papier présentant ODD, l'ensemble des dix éléments ne doit pas être obligatoirement utilisé, seuls les éléments pertinents pour le modèle doivent apparaître.

4.2.3. *Communication des données en entrée et sortie du modèle*

Dans cette partie doit figurer la description des trajectoires d'états en sortie du modèle, c'est-à-dire calculées soient directement par le modèle, soit par des statistiques à partir des données simulées et utilisées pour évaluer le modèle au regard des objectifs, de la question posée. Ces variables d'états d'intérêt doivent être décrites par ce qu'elle représente dans le système source, leur nature discrète ou continue, leur unité, leur granularité et leur étendue. Les données (ou trajectoires d'état en entrée) doivent être décrite de la même manière que les trajectoires d'état en sortie.

4.3. *Spécification formelle du cadre expérimental du modèle*

Dans cette partie, nous discutons la manière de relier les objectifs de modélisation au cadre expérimental. Les objectifs de modélisation définissent les variables d'états d'intérêt. A partir de ces données, il est possible de définir les « *observables* » du modèle, c'est-à-dire une partie des données effectivement calculées et que l'on décide de stocker (ou de visualiser). Ces observables sont ensuite reliées au variables d'état du modèle qui sont à leur tour reliées entre elles par la structure du modèle. Le cadre expérimental doit également spécifier le type des trajectoires d'états en entrée du modèle. Ces différentes trajectoires spécifient les différents contextes dans lequel le modèle peut être utilisé. Tous les cadres doivent être spécifiés formellement de la même manière. Pour cela, nous proposons l'utilisation du formalisme développé par Traoré et Muzzy [TM06], nous ne le reprenons pas ici et renvoyons le lecteur vers l'article original. L'important ici est de savoir que ce formalisme permet une spécification non ambiguë des cadres expérimentaux aux différents niveaux focaux.

4.4. *Spécification formelle du modèle*

Nous distinguons ici les modèles composés des modèles élémentaires. L'une ou l'autre des parties doit être utilisée en fonction de l'itération dans laquelle se situe la spécification.

4.4.1. *Spécification des modèles élémentaires*

Un modèle élémentaire est mono-formalisé. Il convient donc de décrire le modèle dans le formalisme approprié (évènements discrets, équation différentielle, automates, réseaux de Petri, logique propositionnelle, etc.). Si le modèle ne peut être décrit formellement, il conviendra de donner les algorithmes de résolution. Les données en entrée et sortie du modèle doivent être identifiées, ainsi que les observables. Nous préconisons ici le formalisme DEVS (Discrete EVent Systems specification) [ZKP00].

4.4.2. *Spécification des modèles composés ou couplés*

La composition des modèles couplés doit être décrite formellement en utilisant des structures qui décrivent le graphe de modèles couplés et les interfaces en entrée et sortie des modèles. Nous préconisons ici encore le formalisme DEVS pour les modèles couplés. Il est en effet nécessaire que le couplage vérifie la propriété de fermeture dans notre méthode, c'est le cas de DEVS.

4.5. *Spécification formelle du simulateur et du coordinateur associé*

Comme pour les modèles, les simulateurs sont divisés en deux types. Un type pour la simulation des modèles élémentaires et l'autre pour les modèles couplés, appellés alors coordinateurs. La figure 3 illustre la correspondance entre les hiérarchies de modèles et de simulateurs et coordinateurs. Les simulateurs échangent des évènements via les coordinateurs. Le coordinateur racine joue le rôle de chef d'orchestre en envoyant les ordres d'execution aux simulateurs, et donc en assurant la synchronisation.

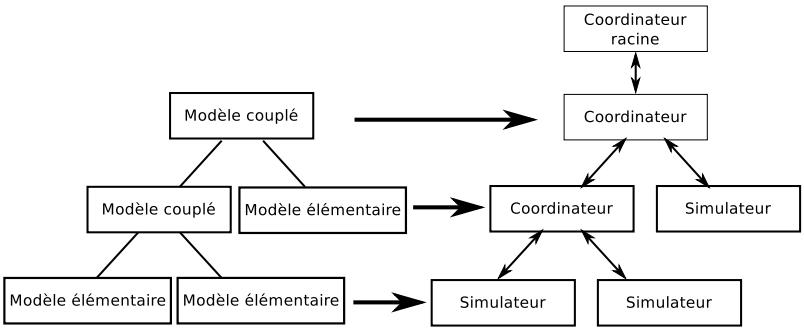

Figure 3 – Correspondance entre la hiérarchie de modèles élémentaires et de modèles couplés avec respectivement la hiérarchie des simulateurs et des coordinateurs associés (repris de Zeigler [ZKP00]. Les doubles flêches figurent les échanges d'évènements qui existent entre les simulateurs via les coordinateurs.

De la même façon que pour la spécification des modèles, la spécification des simulateurs ou des coordinateurs doit être effectuée en fonction de l'itération dans laquelle se situe la spécification.

4.5.1. *Spécification des simulateurs de modèles élémentaires*

Pour spécifier les simulateurs de modèles élémentaires, la TMS introduit la notion de simulateur abstrait, qui décrit sous forme algorithmique un comportement dynamique. Par exemple les algorithmes d'intégration numérique des équations différentielles sont considérés comme des simulateurs abstraits dans le cadre de la TMS. Leur communication peut se faire sans ambiguïté. Les simulateurs abstraits définissent une ou plusieurs fonctions dont la signature est unique (le type et le nombre d'argument est unique). Les signatures des fonctions jouent le rôle d'interfaces avec l'extérieur du modèle élémentaire. L'ordre d'appel de ces fonctions est également unique et toujours le même (la suite de calculs effectués par un intégrateur d'équations différentielles ne peut pas être implémentée dans un ordre aléatoire et toujours différent).

4.5.2. *Spécification des simulateurs de modèles couplés*

Les informations échangées par les modèles sont gérées par un coordinateur qui correspond à la spécification opérationnelle du modèle couplé. Le coordinateur assure le routage des informations entre modèles de même niveau et entre modèles de niveau hiérarchiques $N + 1$ et $N - 1$. Au niveau le plus élevé, on trouve le coordinateur racine qui effectue l'appel des fonctions de chaque modèles composants dans un ordre déterminé. Le choix du composant imminent et le résultat d'un calcul fait au niveau des modèles élémentaires. Ainsi, l'avancement du temps est explicite au niveau modèle, le simulateur correspond à l'implémentation des choix fait par le modélisateur et n'impose pas de choix d'ordonnancement (voir [ZKP00] pour les détails). À chaque formalisme correspond un simulateur abstrait, le couplage des simulateurs abstraits réalise un simulateur multi-formalisme.

Nous n'entrons pas ici dans les détails de cette mécanique. Nous proposons de spécifier pour chaque modèle composant, un simulateur abstrait. Ainsi, les architectures et algorithmes des coordinateurs étant

connus, la description de l'ordonnancement du temps dans le modèle global devient transparente et non ambiguë. Il est définit au niveau des coordinateurs, indépendamment des modèles.

Cette spécification issue de la TMS permet de limiter les erreurs de correspondance entre modèle et simulateur. Le livre de Zeigler et al. [ZKP00] décrit notamment comment simuler de nombreux formalismes en implémentant des simulateurs abstraits en DEVS. Ainsi la preuve que le simulateur est homomorphe avec les modèles qu'il implémente peut être trouvée dans la littérature. Pour résumer, si le simulateur est spécifié et cette spécification référencée, alors il est possible de s'y référer pour toute implémentation future, ce qui allège la spécification complète des modèles.

4.6. *Communication des expériences de simulation*

Une expérience met en relation les bases de cadres expérimentaux, de structure et de modèle. Elle est stockée dans la base expérience de dimulation de la figure 2. Un expérience utilise une structure, donne des valeurs aux paramètres et aux états initiaux des modèles en respectant les cadres expérimentaux associés et simule le modèle composé pour une durée définie (donnée a priori ou calculée par le modèle). Un plan d'expérience correspond bien à la génération d'expériences successives selon une politique donnée [STCR04]. Nous pensons tout comme Grimm et al. [GUD+10] que le plan d'expériences doit être dissocié de la présentation du modèle et faire l'objet d'une partie indépendante de la description du modèle.

Une fois que les quatre bases présentées figure 2 sont spécifiées, il est possible selon nous de discuter de façon beaucoup plus claire et crédible des résultats observés lors des plans d'expériences qui vont associés des structures, des modèles et des cadres expérimentaux. La façon de décrire ces plans d'expériences est libre. Néanmois elle doit être complète (description des plages de valeurs prisent par les paramètres, graine du générateur aléatoire, valeurs initiales des variables d'états, références à la base de cadres expérimentaux et à la base de structures).

La figure 4 fait le lien entre les quatre bases de la figure 2 et les cinq points de la méthode que nous proposons. Une fois les différents points publiés pour un niveau focal donné, il est possible de se référer aux différentes bases pour obtenir les spécifications formelles et opérationnelles des modèles. La composition de modèles nécessite de repasser par l'ensemble des cinq points uniquement si l'on ajoute de nouveaux modèles élémentaires au nouveau niveau focal.

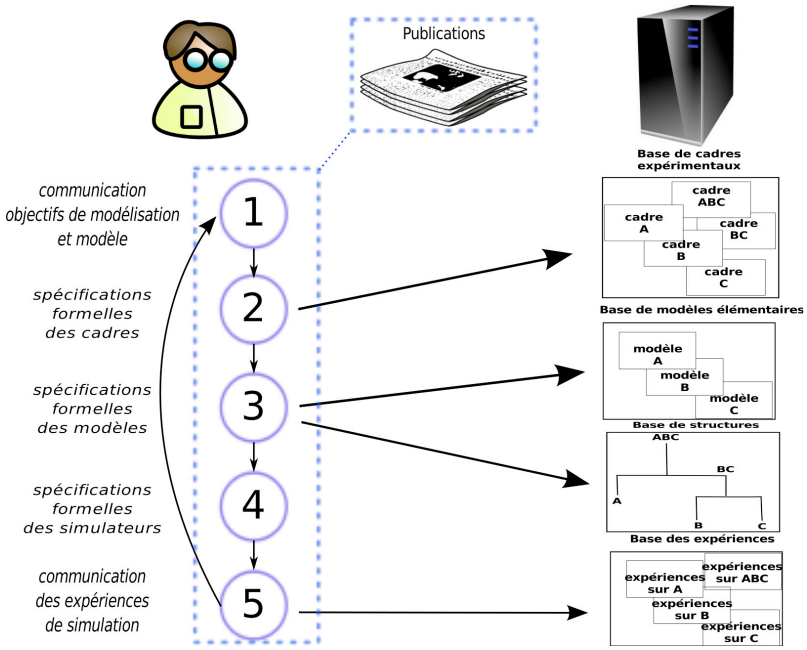

Figure 4 – Correspondance entre l'itération formée par les cinq points de la méthodologie et les quatre bases de notre proposition. Seuls les poins 2, 3 et 5 donnent lieu a un enregistrement dans une base qui permet la réutilisation concrète des spécifications. L'ensemble des points doit faire l'objet de publication avant que les bases soient renseignées.

5. Discussion

Ce qui a motivé ce travail est un essai de rapprochement de deux communautés qui s'ignorent presque totalement, nous les avons appelées communauté ABS et TMS dans ce papier. Une des grandes différences entre ces communautés concerne les objectifs de modélisation. Il n'y a bien sûr pas de limite de séparation nette entre les deux communautés, néanmoins nous pouvons dire que la communauté ABS va davantage être orientée vers la caractérisation, la description et l'explication des processus qu'elle modélise de manière à augmenter la connaissance sur ces processus, à faire des prédictions, etc., alors que la communauté TMS va plutôt essayer de reproduire le comportement des systèmes qu'elle modélise à des fins d'optimisation et de conception. Ces deux usages de la modélisation peuvent être basés sur les mêmes outils de conception et de communication des modèles de simulation. C'est l'idée que nous défendons ici. De plus, ces deux usages se confondent lorsqu'il s'agît d'optimiser des décisions de gestion de systèmes socio-environnementaux par exemple, ce qui est une demande forte du coté simulation actuellement. Nous pensons que la méthode proposée ici permet d'intégrer ces deux usages de façon rigoureuse, dans une même spécification.

Notre proposition considère donc deux niveaux dans la communication des modèles. Le premier est le positionnement du modèle dans son contexte d'utilisation (question de recherche, d'ingénierie) en explicitant les concepts, les hypothèses méthodologiques qui justifient la conception du modèle. Le deuxième niveau concerne la spécification formelle des modèles. Ces deux niveaux sont naturellement interdépendants. La communauté d'utilisateurs de modèles ABS nous semble avoir répondu au premier niveau. Tandis que la communauté en TMS nous semble bien répondre au second.

Nous ne proposons pas une méthode pour tester la validité du modèle au regard de la question de recherche posée, mais bien une méthode de communication, de spécification des modèles de simulation à des fins de vérification. Nous n'avons volontairement pas présenté les aspects formels de notre méthode. Nous pensons qu'une description plus textuelle permet d'ouvrir notre réflexion à l'ensemble des utilisateurs de

la modélisation et de la simulation qui ne sont pas forcement des spécialistes de sa théorisation. Néanmoins, l'ensemble des aspects formels qui soutiennent notre méthode sont accessibles dans la littérature citée. Bien sûr, le travail de formalisation complet de notre méthode, présenté dans un seul document, devra être fait pour qu'elle puisse être d'abord testée, puis utilisée.

La relation entre le cadre expérimental et la question posée au modèle (les objectifs) est au cœur de notre méthode. C'est ce lien rendu explicite qui permet la communication rigoureuse du contexte d'utilisation du modèle et qui permet, avec la spécification du modèle lui-même, sa vérification et son utilisation par des tiers. Nous associons donc une base de cadre expérimentaux à la base de structures et à la base de modèles. Notre méthode se base sur une composition hiérarchique de cadre expérimentaux, comme proposé par G. Aumann [Aum07]. Un travail formel reste à faire à ce niveau car Aumann ne propose pas cette composition avec son formalisme. Néanmoins, la solution proposée par Traore et Muzy [TM06] peut servir de base à une telle formalisation. Dans ce contexte, nous pensons que notre méthode ouvre également des perspectives intéressantes sur l'automatisation de la construction de modèles couplés et la vérification de la validité du couplage à des niveaux sémantiques élevés. Des bibliothèques de cadres expérimentaux existent déjà et sont mises à disposition par des outils spécifiques [QDR09], ou même sur internet [CW09]. Pour l'instant, les implémentations dépendent des outil et il n'y a pas de moyen standard de les communiquer malgré les bases formelles qui les sou-tendent.

Dans notre proposition, la spécification du modèle global consiste en une structure choisie dans la base des structures possibles (telles que présentée figure 2) associée à la spécification des modèles composants. Cette solution nous permet de structurer notre modèle par niveaux hiérarchiques sur lesquels nous pouvons nous poser des questions spécifiques à travers des cadres expérimentaux qui sont définis pour chaque niveau. C'est là un apport important de la spécification des modèles complexes puisqu'en précisant l'argumentation sur un niveau hiérarchique, elle augmente la confiance que l'on a à propos des réponses formulées à ce niveau.

Cet article présente rapidement la notion de simulateur abstrait. Ces derniers sont écrits sous forme de pseudo-code. Le pseudo-code n'est pas exempt d'ambiguïté. Prenons un exemple très simple, l'écriture d'une boucle, nous pouvons proposer deux pseudo-codes qui produisent le même résultats :

```
pour tout élément i de la liste
  afficher i
fin pour
```

```
pour i = 1 jusqu'à la longueur de la liste
  afficher le ième élément de la liste
fin pour
```

Le deuxième exemple peut être considéré comme plus précis que le premier, car il laisse moins de liberté quant au choix d'implémentation pour l'algorithme de parcours de la liste. Il est possible de laisser plus ou moins de liberté au programmeur. Néanmoins dans les deux cas, les transitions d'états au niveau modèle sont homomorphiques avec celles du niveau simulateur abstrait, c'est cet homomorphisme qu'il faut prouver.

La dernière étape non discutée dans notre proposition est l'implémentation des simulateurs abstraits dans un langage particulier. Les techniques de vérifications qui s'appliquent ici sont celles du génie logiciel. L'ultime transformation du modèle en code exécutable est la phase de compilation ou d'interprétation du code. Cette phase est considérée comme non ambiguë car très standardisée (normes ISO), même si des choix d'optimisation, sur la représentation des réels par exemple, peuvent rester obscures et faire varier les résultats de calculs sur les flottants.

6. Conclusion

Notre proposition méthodologique pour la spécification de modèles de simulation de systèmes complexes repose principalement sur les travaux de B. Zeigler [ZKP00], G. Aumann [Aum07], Traore et Muzy

[TM06] et V. Grimm et al. [GUD+10]. Notre méthode peut être considérée comme un essai de synthèse et d'intégration de ces travaux. Il est encore nécessaire de l'affiner et de la tester. Nous avons commencé à implémenter cette méthode dans l'environnement de modélisation et simulation « *Virtual Laboratory Environment* » (VLE) [QDR09]. Toutefois, sa complexité limite encore son usage et un effort de clarification doit être entrepris.

La définition d'une méthode rigoureuse et partagée de construction de modèles pour l'étude des systèmes complexes est en construction depuis au moins une quarantaine d'année. Le livre de G. Braziller « *Hierarchy Theory, The Challenge of Complex Systems* » de 1973 est un bon exemple des débats qui animaient déjà les scientifiques de l'époque à ce sujet [Bra73]. La première phrase de la préface du livre de B. Braziller est la suivante :

> « *We are well into a decade in which man can no longer evade the responsability for his own survival. [...] It is a question of survival [...] in a society dominated by increasingly complex local constraints, but lacking stability and rational control.* ».

Trente huit ans plus tard, l'urgence nous impose qu'un protocole de spécification des modèles complexes soit finalisé. Ce protocole reste à écrire dans les détails. Nous espérons que cet article est un pas de plus dans la bonne direction.

Références

[Aum07] G.A. Aumann. A methodology for developing simulation models of complex systems. *Ecological Modelling*, 202 :385–396, April 2007.

[BCK09] J. H. Byun, C. B. Choi, and T. G. Kim. Verification of the devs model implementation using aspect embedded devs. In *Proceedings of the 2009 Spring Simulation Multiconference*, San Diego, USA, 2009.

[Bra73] G. Braziller. *Hierarchy Theory, The Challenge of Complex Systems*. Howard H. Pattee, 1973.

[CW09] Rachid Chreyh and Gabriel Wainer. Cd++ repository : an internet based searchable database of devs models and their experimental frames. In *Proceedings of the 2009 Spring Simulation Multiconference*, SpringSim '09, pages 159 :1–159 :8, San Diego, CA, USA, 2009. Society for Computer Simulation International.

[DRQ04] R. Duboz, E. Ramat, and G. Quesnel. Systèmes multi-agents et théorie de la modélisation et de la simulation : une analogie opérationnelle. In *Actes des douzièmes Journées Francophones sur les Systèmes Multi-Agents (JFSMA) - Systèmes multi-agents défis scientifiques et nouveaux usages*. Olivier Boissier et Zahia Guessoum eds., 2004.

[Fei54] J.K. Feibleman. Theory of integrative levels. *Br. J. Philos. Sci.*, 5(17) :59–66, 1954.

[GUD+10] V. Grimm, Berger U, DeAngelis DL, Polhill G, Giske J, and Railsback SF. The odd protocol : a review and first update. *Ecological Modelling*, 221 :2760–2768, 2010.

[Hah08] C. Hahn. A domain specific modeling language for multiagent systems. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1. AAMAS*, 2008.

[Kli85] G.J. Klir. *Architecture of Systems Complexity*. Sauders, New York, 1985.

[LW05] Y. Labiche and G. Wainer. Towards the verification and validation of devs models. In *in Proceedings of 1st Open International Conference on Modeling and Simulation*, pages 295–305, 2005.

[PTW07] L. Padgham, J. Thangarajah, and M. Winikoff. Auml protocols and code generation in the prometheus design tool.

In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems (AAMAS)*, New York, USA, 2007.

[QDR09] Gauthier Quesnel, Raphaël Duboz, and Éric Ramat. The Virtual Laboratory Environment – An operational framework for multi-modelling, simulation and analysis of complex dynamical systems. *Simulation Modelling Practice and Theory*, 17 :641–653, April 2009.

[SDS10] A. Sturm, D. Dori, and O. Shehory. An object-process-based modeling language for multiagent systems. *Systems, Man, and Cybernetics, Part C : Applications and Reviews*, 40(2) :227–241, March 2010.

[STCR04] Andrea Saltelli, Stefano Tarantola, Francesca Campolongo, and Marco Ratto. *Sensitivity Analysis in Practice : A Guide to Assessing Scientific Models*. Halsted Press, New York, NY, USA, 2004.

[TDZ08] J.P. Treuil, A. Drogoul, and J.D. Zucker. *Modélisation et simulation à base d'agents. Exemples commentés, outils informatiques et questions théoriques*. DUNOD, 2008.

[TM06] Mamadou K. Traore and Alexandre Muzy. Capturing the dual relationship between simulation models and their context. *Simulation Modelling Practice and Theory*, 14 :126–142, 2006.

[Var06] F. Varenne. *Les notions de métaphore et d'analogie dans les épistémologies des modèles et des simulations*. Petra, 2006.

[VUF+06] Grimm V, Berger U, Bastiansen F, Eliassen S, Ginot V, Giske J, Goss-Custard J, Grand T, Heinz S, Huse G, Huth A, Jepsen JU, Jørgensen C, Mooij WM, Müller B, Pe'er G, Piou C, Railsback SF, Robbins AM, Robbins MM, Rossmanith E, Rüger N, Strand E, Souissi S, Stillman RA, Vabø R, Visser U, and DeAngelis DL. A standard protocol for

describing individual-based and agent-based models. *Ecological Modelling*, 198 :115–126, 2006.

[Wai10] G. Wainer. *Discrete-event Modeling and Simulation : Theory and Applications*. CRS Press Inc, 2010.

[Wil99] U. Wilensky. Netlogo. center for connected learning and computer-based modeling, northwestern university. evanston, il., 1999.

[Zei76] B. P. Zeigler. *Theory Of Modeling and Simulation*. Wiley Interscience, 1976.

[ZKP00] B. P. Zeigler, D. Kim, and H. Praehofer. *Theory of modeling and simulation : Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, 2000.