Vers une sémantique des jeux pour un langage d'ingénierie des exigences par buts et agents

Christophe Chareton, Julien Brunel, David Chemouil

Vers une sémantique des jeux pour un langage d'ingénierie des exigences par buts et agents

Christophe Chareton Julien Brunel David Chemouil
Onera – The French Aerospace Lab
F-31055 Toulouse, France
firstname.lastname@onera.fr

Résumé

Cet article propose des structures d'interprétation pour le langage KHI, un langage de modélisation pour l'ingénierie des exigences, qui formalise notamment les notions d'agents, de buts et d'opérations. Deux notions d'agents s'y confrontent : les acteurs, qui sont les agents présents et décrits par leurs capacités d'action sur le système, et les rôles, qui décrivent le comportement attendu des agents pour la réalisation des buts. Les rôles sont ensuite assignés à des acteurs ou des coalitions d'acteurs. La capacité effective des coalitions à jouer les rôles qui leur sont assignés fournit donc un critère de correction du modèle, on appelle le problème relatif *problème de l'assignation*. KHI a une sémantique formelle dans un fragment de la logique temporelle multi-agents ATL*, nommé ATL_{KHI}. Nous décrivons les modèles sémantiques qui permettent d'interpréter ATL_{KHI} et de réduire le problème de l'assignation à un problème de model-checking.

1 Introduction

L'ingénierie des exigences (RE) est la branche de l'ingénierie système qui analyse le domaine d'un problème et s'attache à déterminer les exigences pour un système à développer [13]. Il s'agit de mener l'étude préparatoire à l'élaboration d'un système, de l'identification d'un besoin aux spécifications particulières du système à élaborer. Cette analyse passe notamment par une étude des exigences à remplir.

Cet article poursuit le travail de présentation, initié dans [4], du langage KHI, un langage conçu notamment pour prendre en charge ce qui est identifié comme le *problème de l'assignation*. En RE, les exigences sont assignées à des agents qui sont responsables de leur satisfaction. Le problème de l'assignation est celui de la capacité effective des agents à remplir ces exigences. KHI est formalisé dans la logique multi-agents ATL_{KHI}, un fragment de la logique plus connue ATL* [1].

Cet article décrit les modèles d'interprétation d'ATL_{KHI}. Il s'organise comme suit : la section 2 propose un aperçu de la modélisation par buts et agents pour

l'ingénierie des exigences et fait émerger le problème de l'assignation. Nous présentons en section 3 les éléments du langage KHI puis, en section 4, leur traduction en ATL_{KHI}. La section 5 décrit les modèles CGS_{KHI}, qui donnent une interprétation sémantique des agents de KHI. Nous discutons notre formalisme en section 6 et envisageons en section 7 des directions de futur travail.

2 Ingénierie des exigences

2.1 Buts et opérations

Un *but* est un énoncé décrivant le comportement attendu du système. Dans la méthode KAOS notamment [8, 10, 13], les buts sont progressivement raffinés en buts plus précis. Les buts feuilles à la charge du logiciel à concevoir sont appelés des *exigences*. Ces dernières sont réalisées par des spécifications *d'opérations* et leur réalisation est confiée à des *agents*.

Les buts sont formalisés en logique temporelle linéaire (LTL) [12], qui permet de raisonner sur des séquences discrètes de temps. Les opérateurs de temps utilisés dans LTL sont $\circ$ et $\mathbf{U}$. Intuitivement, ils signifient :

- $\circ\varphi$ que la condition φ est satisfaite dans l'état qui suit l'état courant.
- $\varphi_1 \mathbf{U} \varphi_2$ que la condition φ_1 est satisfaite dans l'état courant et reste vraie jusqu'à ce que la condition φ_2 soit satisfaite.

Définition 1. *Le langage LTL est défini par la grammaire suivante :*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \circ\varphi \mid \varphi \mathbf{U} \varphi$$

où p appartient à un ensemble dénombrable P de propositions atomiques.

LTL s'interprète dans ($\mathbb{N}$, V) où $V : \mathbb{N} \to \mathcal{P}(P)$ est une fonction qui a tout n associe l'ensemble des propositions atomiques qui sont vraies en l'état n.

Définition 2 (Sémantique de LTL)**.** *Soit $V : \mathbb{N} \to \mathcal{P}(P)$ et soit $i \in \mathbb{N}$, alors :*

- $V, i \models_{LTL} p$ *ssi* $p \in V(i)$*, pour tout* $p \in P$ *;*
- $V, i \models_{LTL} \neg\varphi$ *ssi* $V, i \not\models \varphi$ *;*
- $V, i \models_{LTL} \varphi_1 \wedge \varphi_2$ *ssi* $V, i \models_{LTL} \varphi_1$ *et* $V, i \models_{LTL} \varphi_2$ *;*
- $V, i \models_{LTL} \circ\varphi$ *ssi* $V, i+1 \models_{LTL} \varphi$ *;*
- $V, i \models_{LTL} \varphi_1 \mathbf{U} \varphi_2$ *ssi*
  $\exists j \in V (V, i+j \models_{LTL} \varphi_2$ *et* $\forall k < j (V, i+k \models_{LTL} \varphi_1))$ *;*

On utilise aussi les abréviations $\Diamond\varphi := \top \mathbf{U} \varphi$ et $\Box\varphi := \neg\Diamond\neg\varphi$.

On suit ici l'exemple d'une interface de vente en ligne. Dans cet exemple, une librairie a pour but de vendre le roman *Guerre et Paix*, qu'on formalise par : sell"WarAndPeace" : $\Diamond(sentMoney = item.price)$. Ce but est rempli si et seulement si la somme d'argent correspondant au prix du roman est envoyée au cours de l'exécution.

publishAd.$domPre$	:=	$\top$
publishAd.$domPost$	:=	$bestSellerOffer < 30$
$reqTrig$ pour ***simpleTransaction***	:=	$\top$
$reqPost$ pour ***interestingPrice***	:=	$bestSellerOffer < 10$
$reqPost$ pour ***deliveryDelay***	:=	$bestCommittedDeliveryDelay < 5$

FIGURE 1: L'opération *publishAd* et sa spécification

Les buts sont réalisés par des opérations. Celles-ci sont identifiées par une précondition domPre et une post-condition domPost : l'opération est déclenchée dans tout état s tel que domPre est satisfaite en s et domPost est satisfaite dans l'état suivant. Chaque opération est aussi spécifiée par des conditions requises qui contraignent la manière dont elle doit être déclenchée pour satisfaire les exigences. Une condition requise est donc à la fois relative à l'opération qu'elle spécifie et à l'exigence pour laquelle elle est requise [8–10, 13]. Ces conditions sont de trois types dans KAOS :

- Des préconditions nécessaires de déclenchement, reqPre : l'opération n'a pas lieu en l'état s sans que s ne satisfasse reqPre.
- Des préconditions suffisantes de déclenchement, reqTrig : l'opération est déclenchée en tout état où reqTrig ainsi que domPre est satisfaite.
- Des postconditions, reqPost : l'opération n'a pas lieu en l'état s sans que l'état successeur de s ne satisfasse reqPost.

La spécification des opérations pour la réalisation des exigences est appelé *l'opérationnalisation*. La sémantique des opérations et de leurs conditions est donnée par :

Définition 3. *Soit op une opération, alors :*

Opération :	$[\![op]\!]$	$:= op.domPre \wedge \circ op.domPost$
ReqPre :	$[\![op.reqPre]\!]$	$:= \Box([\![op]\!] \rightarrow op.reqPre)$
ReqPost :	$[\![op.reqPost]\!]$	$:= \Box([\![op]\!] \rightarrow \circ op.reqPost)$
ReqTrig :	$[\![op.reqTrig]\!]$	$:= \Box((op.domPre \wedge op.reqTrig) \rightarrow [\![op]\!])$

La figure 1 donne les conditions identifiant et spécifiant l'opération de publication d'une annonce *publishAd* dans notre exemple : $bestSellerOffer < 10$ est ainsi une postcondition requise de l'opération *publishAd* pour *interestingPrice*. On utilisera désormais les abbréviations bSO et $bCDD$ pour les noms *bestSellerOffer* et *bestCommittedDeliveryDelay*. Ces variables sont en fait des abbréviations pour les minima des offres et délais de livraisons par les acteurs du modèle. Nous en donnons l'expression complète dans la section 3.

Remarque. *On notera que les pré- et post-conditions font référence à des variables, qui ne figurent pas dans la définition de LTL stricto sensu. Une présentation plus rigoureuse d'une LTL avec contraintes sur des variables figure dans la section 4, qui présente la sémantique de* KHI.

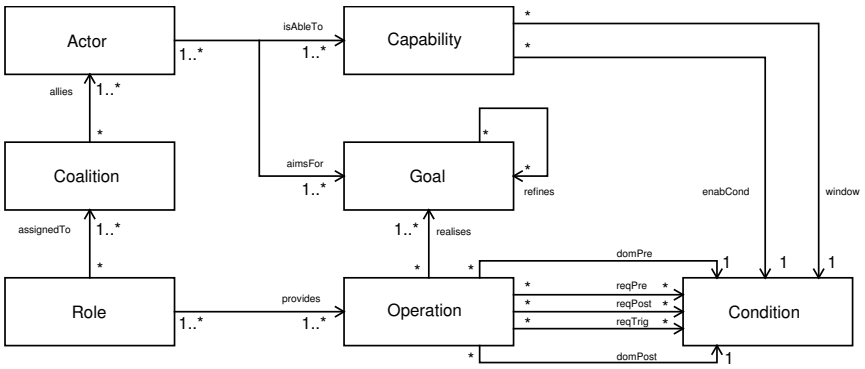

FIGURE 2: Méta-modèle du langage KHI

2.2 Agents

Le concept d'agents est particulièrement important dans les méthodes *i** [14, 15] et TROPOS [2]. On y distingue deux notions d'agents. Les premiers sont les *acteurs* et forment une notion descriptive : ce sont les agents présents dans le système. Ils sont caractérisés à la fois par les buts qu'ils poursuivent et par un ensemble de *capacités*, qui caractérisent leur capacité à influencer l'exécution du système.

La seconde notion est celle de *rôle*, elle est prescriptive. Un rôle est un ensemble d'actions rassemblées par l'ingénieur (selon des critères de pertinence qui n'entrent pas dans le sujet de cet article). C'est donc une description du comportement attendu d'un acteur pour la réalisation des buts. Certaines extensions de TROPOS [5–7, 11] envisagent la relation entre ces deux notions. Elles s'intéressent notamment à la question de la capacité des acteurs à jouer les rôles qui émergent de l'analyse des exigences et de leurs spécifications. Mais ce problème y est exprimé en logique propositionnelle.

3 Le langage KHI

Le métamodèle de KHI est donné dans la figure 2. Les acteurs sont les agents disponibles. Le modèle de buts pour notre cas d'étude est donné dans la figure 3. Comme dans TROPOS, les acteurs y sont représentés dans des ellipses en pointillés indicées par des cercles. Chaque ellipse contient l'ensemble des buts pour l'acteur considéré. Les buts sont représentés par des parallèlogrammes et structurés par la relation de raffinement. Par exemple, le but *meetSupplyAndDemand* pour le site internet est raffiné en *firstHand* et *secondHand*. Les exigences sont ensuite réalisées par des opérations qui sont identifiées et spécifiées de la même manière que dans KAOS. En toute rigueur, les conditions $reqPre$, $reqPost$ et $reqTrig$ devraient être une classe d'association liée à *realises*. Les opérations sont représentées dans la figure 4 par des hexagones. Elles sont rassemblées dans des rôles (les ellipses, indicées par des cercles soulignés).

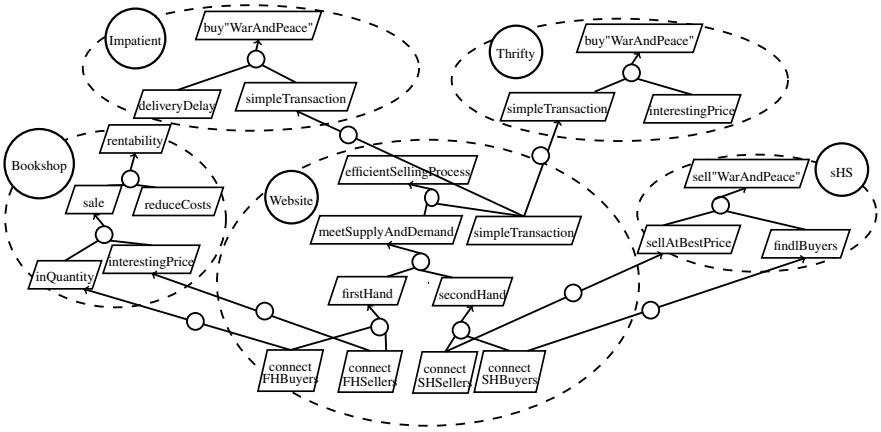

FIGURE 3: Acteurs et buts pour l'interface de vente en ligne

bookshop	*wait*	$enabCond := \top$ $window := \top$
	proposePrice	$enabCond := \top$ $window :=$ *1sO* $\in [15, 30]$
	commitToDeliver	$enabCond := \top$ $window :=$ *1cDD* $\in [3, 10]$
sHS	*wait*	$enabCond := \top$ $window := \top$
	proposePrice	$enabCond := \top$ $window :=$ *2sO* $\in [5, 15]$
	commitToDeliver	$enabCond := \top$ $window :=$ *2cDD* $\in [8, 15]$

TABLE 1: Capacités de la librairie et du vendeur de seconde main

Les acteurs ont aussi des capacités, qui sont décrites par une précondition et une fenêtre. Dans tout état qui satisfait la précondition, l'agent correspondant contrôle les variables de la fenêtre dans les limites qu'elle définit. Dans le tableau 1 les variables $1sO$ et $2sO$ représentent respectivement les offres de prix par *bookshop* et *secondHandSeller* (*sHS*) et les variables $1cDD$ et $2cDD$ représentent leurs temps de livraison. La fenêtre *1sO* $\in [15, 30]$ par exemple rend la librairie capable de proposer le roman à n'importe quel prix compris entre 15 et 30 euros. Il peut arriver que deux ou plusieurs acteurs aient des capacités concurrentes, celles d'agir sur la même variable à partir du même état. De tels choix contradictoires correspondent à un blocage du système, nous appelons cette situation un *deadlock*.

Nous formalisons bSO comme une expression du minimum des prix proposés. Dès lors toute formule $\varphi[bSO]$ de Cond_{KHI} est une abbréviation pour : $((1sO - 2sO \leq 0) \rightarrow \varphi[1sO]) \wedge ((1sO - 2sO > 0) \rightarrow \varphi[2sO])$. De même $\varphi(bCDD)$ abrège : $((1cDD - 2cDD \leq 0) \rightarrow \varphi[1cDD]) \wedge ((1cDD - 2cDD > 0) \rightarrow$

$\varphi[2cDD])$.

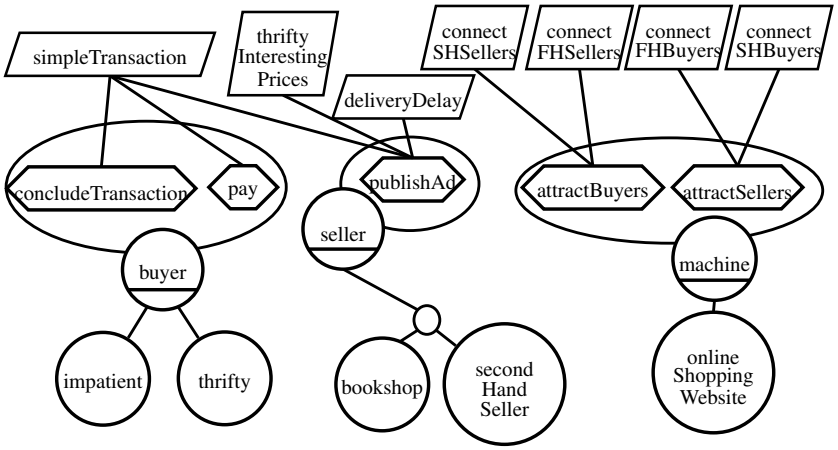

FIGURE 4: Buts feuilles, opérations, rôles et assignation dans le cas d'étude

Les acteurs peuvent finalement être rassemblés dans des coalitions, auxquelles les rôles sont assignés. Dans la figure 4 par exemple, le rôle *seller* est assigné à la coalition formée des acteurs *bookshop* et *sHS*. On identifie le problème de l'assignation comme celui de la capacité effective de chaque coalition à jouer les rôles qui lui sont assignés.

4 Sémantique

Dans cette section, nous donnons la sémantique du langage KHI dans ATL_{KHI}. Pour ce faire, nous présentons d'abord les outils formels utilisés.

4.1 Pré-requis formels : logiques temporelles et multi-agents

ATL_{KHI} est construite en trois étapes, en intégrant successivement des opérateurs temporels et de choix à la logique propositionnelle.

Tout d'abord un ensemble Cond_{KHI} de conditions, qui sont des combinaisons booléennes de contraintes pour des variables. On l'utilise pour exprimer les conditions *dom* et *req* pour les opérations, et les $enabCond$ et les $window$ pour les capacités des acteurs.

Définition 4. *Soit un ensemble de variables U, le langage Cond_{KHI} sur U est engendré par la grammaire suivante :*

$$\varphi ::= x \sim n \mid x - y \sim n \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi$$

où $x, y \in U$, $n \in \mathbb{N}$, et $\sim \in \{<, >, =, \leq, \geq\}$.

La définition d'une $window$ utilise en fait un fragment de Cond_{KHI} dans lequel les variables sont explicitement bornées. Ce sont donc les conditions du type $a \leq x \wedge x \leq b$ combinées avec les opérateurs $\wedge$ et $\vee$.

On interprète Cond_{KHI} dans des valuations de U dans $\mathbb{N}$. Pour $\varphi \in \text{Cond}_{\text{KHI}}$ et $v \in \mathbb{N}^U$, on note $v \models_{\text{Cond}_{\text{KHI}}} \varphi$ si φ est vraie dans v.

Le deuxième étage est donné par la logique temporelle linéaire dont les atomes sont dans Cond_{KHI} (LTL_{KHI}). Elle sert pour l'expression des buts et l'opérationnalisation.

La logique *Alternating-time Temporal Logic* pour KHI (ATL_{KHI}) permet enfin de décrire les capacités. En logique multi-agents, on considère différentes évolutions possibles du temps. À tout point de l'exécution, plusieurs évolutions possibles du temps sont prises en compte et l'exécution effective est déterminée par les choix des agents. On représente cela par l'opérateur $\langle\langle A \rangle\rangle$, qui signifie que l'agent ou la coalition d'agents A peut assurer la satisfaction de la formule dans sa portée, quoi que fassent les autres acteurs.

Définition 5. *L'ensemble des formules de ATL$_{\text{KHI}}$ est engendré par :*

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle \psi$$

où $p \in$ Cond$_{\text{KHI}}$, $\psi \in$ LTL$_{\text{KHI}}$ et A est un(e) (coalition d') acteur(s).

La sémantique pour ATL* est donnée dans des modèles dénommés *Concurrent Game Structures (CGS) :* ce sont des structures arborescentes dans lesquelles les agents font des choix qui influencent l'exécution. Nous les présentons ici dans leur forme générale. Leur étude particulière pour KHI est l'objet de la section 5.

Définition 6. *Une CGS est un tuple $S = \langle \Sigma, Q, \Pi, \pi, d, \delta \rangle$ où :*
- *Σ est un ensemble d'agents ;*
- *Q est un ensemble d'états, appelé le* domaine *de la CGS ;*
- *Π est un ensemble de propositions atomiques ;*
- *π est une fonction de valuation, de Q dans $\mathcal{P}(\Pi)$;*
- *d est la fonction de choix jouables : $d : \Sigma \times Q \to \mathbb{N}^*$. À l'état courant et à chacun des agents, cette fonction associe le nombre de choix possibles pour l'agent (il en existe au moins un).*
- *δ est la fonction de transition. C'est une fonction partielle de $Q \times \mathbb{N}^\Sigma$ dans Q. Étant donné l'état courant $q \in Q$, et pour chaque agent a, un choix $j_a < d(a, q)$, $\delta(q, \langle j_a \rangle_{a \in \Sigma}) \in Q$ donne l'état successeur.*

La définition de la relation de satisfaction est donnée dans les CGS pour KHI en section 5.

4.2 Sémantique de KHI dans ATL$_{\text{KHI}}$

On traduit maintenant KHI dans LTL_{KHI} et ATL_{KHI}. Tout d'abord, un but est une formule de LTL_{KHI} : $[\![g]\!] \in \text{LTL}_{\text{KHI}}$.

Définition 7 (Sémantique de refines)**.** *La relation $refines$ est telle que la satisfaction de l'ensemble des buts rafinant assure la satisfaction du but raffiné. Soit g un but*[1]*, alors* $\{[\![g.refines^{-1}]\!]\} \models_{LTL_{\text{KHI}}} [\![g]\!]$.

La sémantique des opérations et de leurs spécifications est la même que dans KAOS [8, 10, 13] (cf. section 2.1).

Définition 8 (Sémantique de realises)**.** *La relation $realises$ est telle que la satisfaction de l'ensemble des spécifications des opérations réalisantes assure la satisfaction du but réalisé. Soit g un but, alors :*

$$\{[\![g.realises^{-1}.req]\!]\} \models_{LTL_{\text{KHI}}} [\![g]\!]$$

où $g.realises^{-1}.req$ est l'ensemble des conditions dans $g.realise^{-1}.reqPre$, $g.realises^{-1}.reqPost$ et $g.realises^{-1}.reqTrig$.

Les deux concepts d'agents diffèrent par leurs statuts : les *acteurs* sont décrits dans KHI avec leurs différentes capacités qui se traduisent en modèles sémantiques (cf. section 5). En revanche, un *rôle* est traduit en ATL_{KHI} par l'ensemble des opérations qu'il assure :

Définition 9 (Sémantique des rôles)**.** $[\![rl]\!] := \Box \bigwedge_{r \in req}([\![r]\!])$

où req est l'ensemble des conditions dans $rl.provides.reqPre$, $rl.provides.reqPost$ et $rl.provides.reqTrig$.

On a ainsi pour le rôle *seller* : *seller.provides* = {*publishAd*}. Alors, selon les spécifications données pour *publishAd* dans la Fig. 1 :

$$[\![\textit{seller}]\!] := \Box((\circ(bSO < 10)) \wedge (\circ(bCDD < 5)))$$

Le problème de l'assignation est réduit à celui de la satisfaction de la formule $\langle\langle c.allies\rangle\rangle [\![r]\!]$ dans une structure sémantique $\mathfrak{M}$ qui traduit les capacités des acteurs du système, dont ceux de c. Dans la section suivante, nous décrivons la manière dont un modèle tel que $\mathfrak{M}$ est construit à partir des capacités des acteurs. On donne ici la sémantique de la relation $assignedTo$.

Définition 10 (Sémantique de assignedTo)**.** *Soit r un rôle et $c \in r.assignedTo$ une coalition à laquelle r est assigné. Alors, étant donné un modèle sémantique $\mathfrak{M}$ représentant la capacité de tous les acteurs (y compris ceux de c), c est capable de jouer le rôle r,* i.e.*, $\mathfrak{M} \models_{ATL_{\text{KHI}}} \langle\langle c.allies\rangle\rangle [\![r]\!]$.*

Cette traduction de KHI en ATL_{KHI} s'interpète dans des CGS particulièrement spécifiées pour l'interprétation de KHI. On les appelle CGS_{KHI}. Elles permettront notamment un traitement algorithmique du problème de l'assignation. À partir d'une instance $\mathfrak{K}$ de KHI la section 5 décrit ces structures en deux étapes : la première, $\mathfrak{M}$, est directement engendrée par une description des états possibles du système dans Cond_{KHI}. Et la seconde provient d'un quotient du domaine de $\mathfrak{N}$, est elle-même à domaine fini et rend donc possible la description de procédures de *model-checking* décidables notamment pour le problème de l'assignation.

1. On note R^{-1} l'inverse d'une relation R, $\Gamma \models_{\text{LTL}_{\text{KHI}}} \varphi$ signifie que φ est une conséquence sémantique de Γ.

5 KHI et les CGS_{KHI}

Pour $\mathfrak{M}$ comme pour $\mathfrak{N}$, on définit successivement un ensemble d'agents, un domaine, un ensemble de propositions atomiques, une valuation et des fonctions de choix jouables et de transitions.

Le domaine de $\mathfrak{M}$ est l'ensemble des états possibles, *i.e.* il correspond à l'ensemble des assignations possibles pour les variables dans $\mathfrak{K}$ qui respectent les propriétés du domaine. On obtient le domaine de $\mathfrak{N}$ en fusionnant, parmi les états du premier, ceux qui satisfont les mêmes propositions parmi celles utilisées dans $\mathfrak{K}$.

Dans les deux cas, on ajoute au domaine un état particulier, nommé $\overline{\perp}$, qui correspond à la situation de *deadlock* : cet état a une valuation vide. L'exécution y est conduite dans le cas où des agents expriment des choix contradictoires. L'état $\overline{\perp}$ est son unique successeur. Une exécution qui y parvient ne se poursuit donc plus dans le reste du domaine.

Par leurs choix, les agents décrivent directement le sous-ensemble des potentiels états successeurs qu'ils laissent possible. L'action des agents consiste en effet à déterminer éventuellement la ou les valeurs d'une ou plusieurs variables. Chaque choix pour un agent a est donc représenté par l'ensemble des états compatibles avec ces valeurs. À chaque état, l'ensemble des choix disponibles pour un agent est clos par intersection, de manière à ce qu'il puisse éventuellement y agir selon une ou plusieurs de ses capacités.

La fonction de transition identifie le successeur d'un état en fonction de l'état courant et de l'ensemble des choix exprimés par les agents. Sauf choix contradictoires de la part des acteurs, ce successeur est toujours un élément de l'intersection des choix. Il est déterminé au sein de cet ensemble comme réponse au problème du cadre *(frame problem) :* en l'absence d'un choix explicite exprimé en ce sens par les acteurs, la valeur d'une variable ou la satisfaction d'une condition n'est pas modifiée au cours de l'exécution.

5.1 Une structure infinie : $\mathfrak{M}$

On définit d'abord un modèle issu directement de l'ensemble des états possibles du système. Soit $\mathfrak{K}$ une instance de KHI, *l'ensemble des agents* dans $\mathfrak{M}$ est identifié à l'ensemble des acteurs dans $\mathfrak{K}$: $\Sigma = \mathit{Actors}$.

Le *domaine* M est composé d'un ensemble d'états en bijection avec l'ensemble des assignations possibles des variables utilisées dans $\mathfrak{K}$ et de l'état $\overline{\perp}$. Soit X l'ensemble des variables dans $\mathfrak{K}$, c'est-à-dire l'ensemble des variables ayant au moins une occurence dans l'ensemble *Roles.provides.cond* $\cup$ *Actors.isAbleTo.enabCond*, où *Roles.provides.cond* $=$ *Roles.provides.req* $\cup$ *Roles.provides.domPre* $\cup$ *Roles.provides.domPost*. On distingue un état $\overline{v}$ de $M \setminus \{\overline{\perp}\}$ et l'assignation correspondante v. On obtient ainsi M :

Définition 11 (Domaine M)**.** $M = \{\overline{v} \mid v \in \mathbb{N}^U\} \cup \{\overline{\perp}\}$

Les propositions atomiques de $\mathfrak{M}$ sont l'ensemble des formules dans le langage propositionnel.

Définition 12 (Propositions atomiques)**.** $\Pi_M = \text{Cond}_{\text{KHI}}$

La fonction de valuation est donnée par la satisfaction sémantique.

Définition 13 (Valuation π_M)**.**
- $\pi_M(\overline{\bot}) = \emptyset$
- *Pour tout* $\overline{v} \in M \setminus \{\overline{\bot}\}$ *et* $\varphi \in$ *Cond*$_{\text{KHI}}$, $\varphi \in \pi_M(\overline{v})$ *ssi* $v \models_{Cond_{\text{KHI}}} \varphi$

Les acteurs expriment leurs choix dans KHI en décidant la valeur de certaines variables. On définit la notion de *fibres* pour traduire ces choix dans le modèle. Les fibres sont des ensembles de valuations qui ont en commun la valeur pour une ou plusieurs variables.

Définition 14 (Fibre au-dessus d'une valuation partielle)**.** *Soit* v *une valuation sur un ensemble de variables* X_2 *et soit* X_1 *un ensemble de variables tel que* $X_2 \subset X_1$*. On note alors* $fb_{X_1}(v)$ *et on appelle* fibre sur X_1 au dessus de v *l'ensemble des valuations sur* X_1 *qui étendent* v *:*

$$\mathit{fb}_{X_1}(v) = \{v' : X_1 \to \mathbb{N} \mid \textit{ pour tout } x_i \in X_2, v'(x_i) = v(x_i)\}$$

On note alors $v = v'_{\restriction X_2}$. On appelle par ailleurs *la base de fb*, et on note *base(fb)*, l'ensemble X_2 des variables sur lequel une fibre *fb* est définie. On note encore par la suite $\overline{\mathit{fb}_{X_1}(v)}$ l'ensemble des états dans $\mathfrak{M}$ qui correspondent à des valuations dans $\mathit{fb}_{X_1}(v) : \overline{\mathit{fb}_{X_1}(v)} = \{\overline{w} \mid w \in \mathit{fb}_{X_1}(v)\}$.

Nous arrivons maintenant à la définition de la fonction de choix jouables. On définit d'abord un choix qui correspond à l'exercice d'une capacité pour l'agent a : dans l'état $\overline{v}$, si a a une capacité c telle que $c.enabCond$ est satisfaite en $\overline{v}$, alors il peut donner aux variables dans $c.window$ n'importe quelle valeur qui satisfait $c.window$ en sélectionnant la fibre correspondante. L'ensemble de ces choix est ensuite clos par intersection, de manière à ce qu'un agent puisse jouer selon une, plusieurs ou aucune de ses capacités en même temps.

Définition 15 (Fonction de choix jouables $c_M : (M \times \Sigma) \to \mathcal{P}(\mathcal{P}(M))$)**.** *Soit* $G \subset M$*, pour tout agent* a *et pour tout état* $\overline{v}$*,* $G \in c_M(\overline{v}, a)$ *ssi*
- *Il existe une capacité* $(enabCond, window)$ *pour* a *telle que* $v \models_{Cond_{\text{KHI}}} enabCond$*, une assignation* $w : \text{sg}(window)$ [2] $\to \mathbb{N}$ *telle que* $G = \overline{\text{fb}_X(w)}$ *et pour tout* $w' \in G$*,* $\overline{w'} \models_{Cond_{\text{KHI}}} window$*.*
- *Ou il existe deux choix* $G_1, G_2 \in c_M(\overline{v}, a)$ *pour* a *dans l'état* $\overline{v}$ *et* $G = G_1 \cap G_2$*. Autrement dit,* $c_M(\overline{v}, a)$ *est clos par intersection.*

c_M est étendue aux coalitions d'acteurs : les choix disponibles pour une coalition sont les intersections des choix disponibles pour ses membres :

2. sg(ϕ) est l'ensemble des variables qui apparaissent dans ϕ

Définition 16. *Soit $c = (a_1, \ldots, a_k)$ une coalition d'agents. Alors, pour tout $G \in \mathcal{P}(M)$, $G \in c_M(c)$ ssi il y a $G_1, \ldots G_k$ tels que $G = \bigcap_{i \in [1 \ldots k]}$ et pour tout $i \leq k$, $G_k \in c_M(a_k)$*

En particulier, un agent a peut être décrit avec la capacité $(\varphi, \top)$. Celle-ci lui permet de ne pas agir concrètement sur le système. C'est le cas dans notre exemple pour chacun des acteurs *bookshop* et *sHS*, avec la précondition $\top$. Jouer ce choix correspond à forcer l'exécution dans la fibre à base vide, c'est à dire dans l'ensemble M. Autrement dit, il n'intervient pas sur la suite de l'exécution. On définit maintenant la fonction de transitions.

Définition 17 (Fonction de transition $\delta_{\mathbf{M}}$:)**.** *Étant donné un état $\overline{v} \in M$, et, pour chaque agent $a \in \Sigma$, un choix $m_a \in c_M(\overline{v}, a)$,*

$$\delta(\overline{v}, \langle m_a \rangle_{a \in \Sigma}) = \begin{cases} \overline{\bot} & \textit{si } \overline{v} = \overline{\bot} \textit{ ou } \bigcap\limits_{a \in \Sigma} m_a = \emptyset \\ \bigcap\limits_{a \in \Sigma} m_a \quad \cap \quad \overline{fb_X(v_{\restriction X_{\text{Pass}}})} & \textit{sinon} \end{cases}$$

avec $X_{\text{Pass}} = \{x_k \in X \mid \forall a \in \Sigma x_k \notin \textit{base}(m_a)\}$.

Intuitivement, $\overline{fb_X(v_{\restriction X_{Pass}})}$ représente les assignations pour lesquelles les variables non affectées par les choix des agents (les variables dans X_{Pass}) ont la même valeur que dans l'assignation v, correspondant à l'état courant.

Une stratégie f_a pour un agent $a \in \Sigma$ associe à toute séquence finie d'états σ qui se termine à l'état $\overline{v}$ un choix $f_a(\sigma) \in c_M(\overline{v}, a)$. f_a induit donc un ensemble d'exécutions que l'agent a peut imposer. Une stratégie f_A, pour une coalition d'agents A est définie comme un ensemble de stratégies pour chacun des agents dans A.

$\mathfrak{M}$ est un modèle qui décrit les capacités des acteurs et les états possibles des variables de $\mathfrak{K}$. Schématiquement, $\mathfrak{M}$ est une représentation du *system-as-is* [13] à travers les agents pré-existants. Le problème de l'assignation d'un rôle r à une coalition A dans $\mathfrak{K}$ s'exprime alors par la question de la satisfaction de la formule $\langle\langle A \rangle\rangle r$ dans $\mathfrak{M}$: $\mathfrak{M} \models^{?}_{\text{ATL}_{\text{KHI}}} \langle\langle A \rangle\rangle r$.

Définition 18 ($\models_{\text{ATL}_{\text{KHI}}}$)**.** *Soit $\mathfrak{M}$ une CGS, $v \in M$ un état de $\mathfrak{M}$, A une coalition d'acteurs ψ une formule de LTL$_{\text{KHI}}$ et φ_1 et φ_2 des formules de ATL$_{\text{KHI}}$. Alors :*

- $\mathfrak{M}, s \models_{ATL_{\text{KHI}}} p$ *ssi* $p \in V(s)$
- $\mathfrak{M}, s \models_{ATL_{\text{KHI}}} \neg\varphi$ *ssi il n'est pas vrai que* $M, s \models_{ATL_{\text{KHI}}} \varphi$
- $\mathfrak{M}, s \models_{ATL_{\text{KHI}}} \varphi_1 \wedge \varphi_2$ *ssi* $M, s \models_{ATL_{\text{KHI}}} \varphi_1$ *et* $M, s \models_{ATL_{\text{KHI}}} \varphi_2$
- $\mathfrak{M}, s \models_{ATL_{\text{KHI}}} \langle\langle A \rangle\rangle \psi$ *ssi il y a une stratégie F_A pour A telle que toute exécution $\sigma = s, s_1, \ldots$ compatible avec F_A satisfait ψ ($\sigma, s_0 \models_{LTL_{\text{KHI}}} \psi$)*

Le modèle $\mathfrak{M}$ est cependant de cardinal infini et ne permet pas d'y envisager l'utilisation de procédure finie de *model-checking*. Nous définissons donc une structure finie, $\mathfrak{N}$, qui est le quotient de $\mathfrak{M}$ par l'équivalence obtenue de la satisfaction des sous-ensembles de propositions atomiques.

5.2 Une structure finie : $\mathfrak{N}$

De même que pour $\mathfrak{M}$, nous définissons pour $\mathfrak{N}$ les agents, le domaine N, la valuation π_N et les fonctions de choix jouables c_N et de transition δ_N.

L'ensemble des agents reste l'ensemble des acteurs de $\mathfrak{K}$. La définition du domaine N de $\mathfrak{N}$ nécessite un quotient de l'ensemble M par la relation d'équivalence sémantique modulo les atomes utiles du modèle.

Soit une formule φ de Cond_{KHI}. On note $At(\varphi)$ l'ensemble de ses atomes, *i.e.* l'ensemble de ses sous-formules qui sont de la forme $x \sim n$, où $x \in U$, $\sim \in \{<, >, =, \leq, \geq\}$ et $n \in \mathbb{N}$. Cette notation s'étend à un ensemble de formules Γ : $At(\Gamma) = \bigcup_{\varphi \in \Gamma} At(\varphi)$. On peut alors désigner l'ensemble des atomes des formules utilisées dans les rôles et les préconditions de capacité de $\mathfrak{K}$: $Z = \text{At}(\textit{Roles.provides.cond} \cup \textit{Actors.isAbleTo.enabCond})$.

On définit alors sur M la relation d'équivalence $\equiv_{\mathfrak{N}}$, qui associe deux états si et seulement si ils satisfont les mêmes atomes dans Z.

$$\forall \overline{v}, \overline{v'} \in M(\overline{v} \equiv_{\mathfrak{N}} \overline{v'} \leftrightarrow (\forall p \in Z((\overline{v} \models_{Cond_{\text{KHI}}} p) \leftrightarrow (\overline{v'} \models_{Cond_{\text{KHI}}} p))))$$

Le *domaine* de $\mathfrak{N}$, noté N, est le quotient de M par $\equiv_{\mathfrak{N}}$. On écrit par la suite $\overline{\bot}$ pour la classe d'équivalence contenant $\overline{\bot}$. N est donc en bijection avec un sous-ensemble de $\mathcal{P}(Z)$: à chaque état $\overline{s}$ de N est associé un élément s de $\mathcal{P}(Z)$ qui est l'ensemble des conditions de $At(Z)$ vraies dans toute assignation de $\overline{s}$.

L'ensemble des propositions atomiques de $\mathfrak{N}$ est réduit à celles des propositions de Π_M qui sont présentes dans $\mathfrak{K}$. Il s'agit ainsi d'un ensemble fini : $\Pi_N = Z$.

La *valuation* π_n est donnée de manière immédiate par la correspondance mentionnée ci-dessus entre un état $\overline{s}$ et un ensemble $s \subseteq Z$: pour tout $p \in Z$ et $\overline{s} \in N$, $p \in \pi_N(\overline{s})$ ssi $p \in s$. On a en particulier : $\pi_N(\overline{\bot}) = \emptyset$.

La *fonction de choix jouables* c_N est obtenue en utilisant $\equiv_{\mathfrak{N}}$.

Définition 19 ($c_N : N \times \Sigma \to \mathcal{P}(\mathcal{P}(N))$)**.** *Soit $G \subset N$, alors pour tout agent a et pour tout état $\overline{s}$, $G \in c_N(\overline{s}, a)$ ssi il y a $\overline{v} \in s$ et $S \in c_M(\overline{v}, a)$ tels que $G = [S]_{\equiv_{\mathfrak{N}}}$.*

La *fonction de transition* est définie de manière analogue à δ_M. Si les choix des agents sont non contradictoires, le successeur est pris dans l'intersection des choix des agents. Au sein de cette intersection, s'il reste des atomes dont la valeur de vérité n'est pas fixée, on leur donne la même valeur que dans l'état courant.

Définition 20 (Fonction de transition δ_N)**.** *Étant donné un état $\overline{s} \in N$, et pour chaque agent $a \in \Sigma$, un choix $m_a \in c_N(\overline{s}, a)$, on note $m = \bigcap_{a \in \Sigma} m_a$*

$$\delta_N(\overline{s}, \langle m_a \rangle_{a \in \Sigma}) = \begin{cases} \overline{\bot} \quad \textit{si } m = \emptyset \textit{ ou } \overline{s} = \overline{\bot} \\ m \cap \{\overline{t} \mid \forall p \in Z((\exists \overline{u}, \overline{v} \in m(\overline{u} \models_{Cond_{\text{KHI}}} p, \\ \qquad \overline{v} \not\models_{Cond_{\text{KHI}}} p)) \to ((p \in s) \leftrightarrow (p \in t))\} \textit{ sinon} \end{cases}$$

$\mathfrak{N}$ donne ainsi une interprétation de $\mathfrak{K}$ dans un modèle fini et ouvre la voie pour la résolution du problème de l'assignation. Le théorème suivant, dont la preuve

n'est pas présentée par manque de place, assure l'équivalence, pour les formules avec atomes dans Z, entre les deux structures étudiées dans cette section.

Théorème 21. *Soit $\mathfrak{K}$ un modèle de* KHI*, $\mathfrak{M}$ et $\mathfrak{N}$ les deux structures issues de $\mathfrak{K}$ comme décrit ci dessus, soient $\overline{v} \in M$ et $[\overline{v}]_{\mathfrak{N}}$ la classe d'équivalence de $\overline{v}$ par $\equiv_{\mathfrak{N}}$ et soit φ une formule de ATL$_{\text{KHI}}$ dont les atomes sont dans Z. Alors :*

$$\mathfrak{M}, \overline{v} \models \varphi \textit{ ssi } \mathfrak{N}, [\overline{v}]_{\mathfrak{N}} \models \varphi$$

6 Discussion

KHI englobe à la fois la description dynamique de la réalisation des buts et la mention d'agents avec leurs buts, leurs capacités et leurs interactions.

Ces deux aspects ont déjà été étudiés, mais dans des langages distincts et qui n'offrent pas de sémantique incluant les agents. KAOS propose une sémantique basée sur un temps linéaire : les buts y sont décrits en LTL et les opérations comme des pre et des post conditions. La notion d'agent intentionnel est au coeur de la méthode *i** et la question de la capacité des agents à jouer des rôles a été analysée récemment, notamment en termes d'*engagements* pris par les agents [5–7, 11].

Notre travail vise à unifier ces deux aspects dans un même langage et à enrichir la sémantique pour RE par le problème de l'assignation. Dans KHI, les acteurs décrivent un modèle et leurs capacités à jouer les rôles sont représentées par des formules de ATL$_{\text{KHI}}$. Le problème de l'assignation se traduit ainsi par celui de la satisfaction, au sens de ATL$_{\text{KHI}}$, de la formule pour la capacité à remplir un rôle dans la structure représentant les acteurs.

La capacité des acteurs à jouer des rôles concerne un problème déjà soulevé en RE : celui de l'attribution des spécifications que chaque agent doit assurer. Dans KAOS, il prend la forme de l'assignation de buts aux agents. KAOS ne donne cependant aucun moyen pour discuter la capacité effective des agents à assurer ces spécifications. Une fois raffinés et rassemblés en rôles, les buts sont également confiés à des agents dans TROPOS : la vérification de l'assignation y est également envisagée [5–7], mais formalisée en logique propositionnelle. Elle ignore la dynamique des opérations, qui nous semble essentielle pour la formalisation des interactions entre les agents. Un acteur a_1 peut, par exemple, être capable de réaliser une opération o à condition qu'une condition c soit assurée. Si cette condition est elle-même dépendante d'une opération réalisable par l'agent a_2, alors a_1 et a_2 peuvent ensemble assurer la réalisation de o, à condition de coordonner leurs actions : il faut que a_2 assure la condition c avant que a_1 ne réalise o. Identifier les synergies entre agents et leurs interdépendances nécessite donc une description dynamique des opérations.

7 Conclusion

Dans cet article, nous avons poursuivi le développement du langage KHI en décrivant l'interprétation de son formalisme dans les CGS_{KHI}. Nous disposons donc d'un langage conciliant la description dynamique des opérations et les deux concepts d'agents : les agents intentionnels et les agents prescrits.

La sémantique multi-agents permet de formaliser ces deux notions d'agents et de les confronter sous la forme du problème de l'assignation, traduit en instance du problème de la satisfaction pour ATL_{KHI}. ATL_{KHI} est un fragment de la logique ATL* [1], qui est décidable pour la satisfaction. Le problème de l'assignation est donc ramené à un problème décidable de model-checking.

Cette question de vérification constitue un critère de correction pour les modèles de RE. Le problème de l'existence d'une assignation correcte peut également être traitée. Celui-ci est utile, par exemple, pour identifier les acteurs manquant dans le système : KHI offre des moyens d'expression pour désigner les rôles qui ne sont pas assignables à des acteurs présents. Ces rôles peuvent être perçus comme les spécifications des composants nouveaux à introduire dans le système.

Nous envisageons donc à l'avenir de définir les procédures de décision pour les problèmes de vérification et d'existence d'assignation correcte. Ceci sera l'occasion d'une étude plus précise d'ATL_{KHI}, fragment d'ATL* non comparable avec le fragment mieux connu ATL. Ce travail permettra donc, à partir d'un modèle de KHI décrivant les capacités des acteurs et les exigences, de vérifier une assignation, de décider la possibilité d'une assignation correcte ou, dans le cas négatif, de fournir une spécification précise des agents à introduire dans le système pour la satisfaction des exigences.

Le problème de l'assignation permet de traiter la capacité des acteurs à jouer les rôles qui leur sont assignés. Il ne s'agit cependant pas de leur comportement effectif mais seulement d'un comportement potentiel. Distinguer ces deux notions peut également être intéressant. Cela permettrait par exemple de discuter la cohérence entre un rôle assigné à un agent et la poursuite de ses propres buts, de comparer l'efficacité de différents comportements quant à la satisfaction de certains buts ou d'introduire des conditions fines pour la détermination des rôles, de manière à éviter les cas de *deadlock*.

Une autre direction d'études pourrait donc être l'introduction des stratégies dans le formalisme [3]. Cet ajout dans le langage permettrait en effet d'exprimer distinctement à la fois la capacité d'un acteur à assurer la satisfaction d'un énoncé et son comportement effectif dans le système.

Références

[1] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. In *J. ACM*, pages 672–713, 2002.

[2] Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, and John Mylopoulos. Tropos : An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, pages 203–236, 2004.

[3] T. Brihaye, A. Da Costa, F. Laroussinie, and N. Markey. ATL with strategy contexts and bounded memory. *Logical Foundations of Computer Science*, pages 92–106, 2009.

[4] C. Chareton, J. Brunel, and D. Chemouil. *A Formal Treatment of Agents, Goals and Operations Using Alternating-Time Temporal Logic : 14th Brazilian Symposium on Formal Methods (SBMF 2011), Sao Paulo, Brazil*. 2011.

[5] A.K. Chopra, F. Dalpiaz, P. Giorgini, and J. Mylopoulos. Modeling and reasoning about service-oriented applications via goals and commitments. In *Advanced Information Systems Engineering*, pages 113–128. Springer, 2010.

[6] A.K. Chopra, F. Dalpiaz, P. Giorgini, and J. Mylopoulos. Reasoning about agents and protocols via goals and commitments. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : volume 1-Volume 1*, pages 457–464. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[7] A.K. Chopra and M.P. Singh. Multiagent commitment alignment. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 937–944. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[8] E. Letier. *Reasoning about Agents in Goal-Oriented Requirements Engineering*. PhD thesis, Universite Catholique de Louvain, November 05 2002.

[9] E. Letier and A. van Lamsweerde. Agent-based tactics for goal-oriented requirements elaboration. In *Software Engineering, 2002. ICSE 2002. Proceedings of the 24rd International Conference on*, pages 83 –93, may 2002.

[10] E. Letier and A. Van Lamsweerde. Deriving operational software specifications from system goals. In *Proceedings of the 10th ACM SIGSOFT symposium on Foundations of software engineering*, page 128. ACM, 2002.

[11] A. Mallya and M. Singh. Incorporating commitment protocols into Tropos. *Agent-Oriented Software Engineering VI*, pages 69–80, 2006.

[12] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Compouter Science*, pages 46–57, 1977.

[13] Axel van Lamsweerde. *Requirements engineering, From System Goals to UML Models to Software Specifications*. Wiley, 2009.

[14] E. Yu. Agent-oriented modelling : software versus the world. *Agent-Oriented Software Engineering II*, pages 206–225, 2002.

[15] E. Yu. Social modelling and i*. In *Conceptual Modelling : Foundations and Applications*, 2009.

Annexe : illustration des définitions pour les $\mathbf{CGS_{KHI}}$ à l'aide du cas d'étude

Dans cette annexe on illustre avec le cas d'étude de l'interface de vente en ligne les définitions données dans la section 5 pour les $\mathrm{CGS}_{\mathrm{KHI}}$. On réduit ici le modèle à sa partie engendrée par le rôle *seller* et les acteurs *bookshop* et *sHS*. Les capacités pour ces deux acteurs sont données dans la Fig.1. On rappelle la sémantique du rôle *seller* :

$$[\![\textit{seller}]\!] := \Box((\circ(\mathit{bSO} < \mathit{10})) \wedge (\circ(\mathit{bCDD} < \mathit{5})))$$

Soit :

$$\begin{aligned}[\![\textit{seller}]\!] := \Box(\circ((&\mathit{1sO} - \mathit{2sO} \leq 0) \rightarrow \mathit{1SO} < 10)\\ &\wedge ((\mathit{1sO} - \mathit{2sO} > 0) \rightarrow \mathit{2SO} < 10)\\ &\wedge ((\mathit{1cDD} - \mathit{2cDD} \leq 0) \rightarrow (\mathit{1CDD} < \mathit{5})\\ &\qquad\wedge ((\mathit{1cDD} - \mathit{2cDD} > 0) \rightarrow (\mathit{2CDD} < \mathit{5}))\end{aligned}$$

L'ensemble des variables des préconditions de capacité et des spécifications est donc :

$$U = \{\mathit{1sO}, \mathit{2sO}, \mathit{1cDD}, \mathit{2cDD}\}$$

On décrit dans ce qui suit, élément par élément, les deux CGS pour ce modèle réduit selon les définitions données dans la section 3.

7.1 Le modèle infini $\mathfrak{M}_E$

- Comme expliqué ci-dessus l'ensemble des acteurs pris en compte est $\{\textit{bookshop, sHS}\}$
- Le domaine M : Comme les valeurs de bSO et $fcDD$ sont déterminées par *Dom* en fonction des valeurs des autres variables, M est constitué d'un ensemble en bijection avec $\mathbb{N}^{\{\mathit{1sO},\mathit{2sO},\mathit{1cDD},\mathit{2cDD}\}}$ et de $\overline{\perp}$. On note $\overline{v}$ l'état correspondant ainsi à chaque assignation cohérente v de U dans $\mathbb{N}$
- L'ensemble des propositions atomiques est l'ensemble des formules du langage $\mathrm{Cond}_{\mathrm{KHI}}$ sur U.
- La valuation est immédiate.
- Pour la fonction de choix disponibles : toutes les capacités ont $\top$ pour précondition et on a donc pour tout état $\overline{v} \in M_{ex}$:
  $c_M(\overline{v}, \textit{bookshop}) = \{M_{ex}\}$
  $\cup \{\{\overline{\mathit{1sO} \rightarrow a, \mathit{1cDD} \rightarrow y, \mathit{2sO} \rightarrow z, \mathit{2cDD} \rightarrow t}\}_*\}_{**}$
  $\cup \{\{\overline{\mathit{1sO} \rightarrow x, \mathit{1cDD} \rightarrow b, \mathit{2sO} \rightarrow z, \mathit{2cDD} \rightarrow t}\}_*\}_{**}$
  $\cup \{\{\overline{\mathit{1sO} \rightarrow a, \mathit{1cDD} \rightarrow b, \mathit{2sO} \rightarrow z, \mathit{2cDD} \rightarrow t}\}_*\}_{**}$
  $c_M(\overline{v}, \textit{sHS}) = \{M_{ex}\}$
  $\cup \{\{\overline{\mathit{1sO} \rightarrow x, \mathit{1cDD} \rightarrow y, \mathit{2sO} \rightarrow c, \mathit{2cDD} \rightarrow t}\}_*\}_{**}$
  $\cup \{\{\overline{\mathit{1sO} \rightarrow x, \mathit{1cDD} \rightarrow y, \mathit{2sO} \rightarrow z, \mathit{2cDD} \rightarrow d}\}_*\}_{**}$

$\cup \{\{\overline{1sO \to x, 1cDD \to y, 2sO \to c, 2cDD \to d}\}_*\}_{**}$
où * abrège $x, y, z, t \in \mathbb{N}$ et ** abrège $a \in [15, 30], b \in [3, 10], c \in [5, 15]$ et $d \in [8, 15]$.
Pour la coalition binaire $A = \{bookshop, sHS\}$ on a :

$$\begin{aligned} c_M(\overline{v}, A) = \quad & c_M(\overline{v}, \textit{bookshop}) \\ & \cup\, c_M(\overline{v}, \textit{sHS}) \\ & \cup \{\{\overline{1sO \to a, 1cDD \to y, 2sO \to c, 2cDD \to t}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to a, 1cDD \to y, 2sO \to z, 2cDD \to d}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to x, 1cDD \to b, 2sO \to y, 2cDD \to d}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to x, 1cDD \to b, 2sO \to c, 2cDD \to t}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to a, 1cDD \to b, 2sO \to c, 2cDD \to t}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to a, 1cDD \to b, 2sO \to z, 2cDD \to d}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to a, 1cDD \to y, 2sO \to c, 2cDD \to d}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to x, 1cDD \to b, 2sO \to c, 2cDD \to d}\}_*\}_{**} \\ & \cup \{\{\overline{1sO \to a, 1cDD \to b, 2sO \to c, 2cDD \to d}\}_*\} \end{aligned}$$

– la fonction de transition complète, par la valeur de l'état courant, l'assignation partielle donnée par les choix des acteurs. Soit par exemple l'état $\overline{e}$ correspondant à l'assignation $e := 1sO = 2sO = 40, 1cDD = 2cDD = 20$ et supposons que *bookshop* joue le choix

$$m_b = \{\overline{1sO \to 18, 1cDD \to 5, 2sO \to z, 2cDD \to t}\}_*$$

et que *sHS* joue le choix

$$m_s = \{\overline{1sO \to x, 1cDD \to y, 2sO \to 7, 2cDD \to t}\}_*$$

Alors le successeur de $\overline{v}$ est

$$\delta(\overline{e}\langle m_b, m_s\rangle) := \overline{1sO = 18, 1cDD = 5, 2sO = 7, 2cDD = 20}$$

Les valeurs pour $1sO$ et $1cDD$ sont décidées par m_b, celle pour $2sO$ est décidée par m_s et celle pour $2cDD$ par $\overline{v}$.

7.2 Le modèle fini $\mathfrak{N}_E$

Il est obtenu par le quotient de M par l'équivalence issue de l'égale satisfaction des propositions atomiques utilisées dans le langage.

– l'ensemble des acteurs reste $\{bookshop, sHS\}$
– Pour définir le domaine N, on commence par identifier l'ensemble des propositions atomiques utilisées pour les spécifications et les capacités. Ce sont l'ensemble

$$\begin{aligned} \Pi_N = \{\top, & 1sO - 2sO \leq 0, 1cDD - 2cDD \leq 0, \\ & 1sO < 10, 2sO < 10, 1cDD < 5, 2cDD < 5\} \end{aligned}$$

L'ensemble des états dans $N \setminus \{\overline{\top}\}$ est en bijection b avec celle des assignations de valeurs de vérités aux variables de Π_N qui sont cohérentes. On élimine en particulier toutes les valuations V telles que que $V(\top) = 0$. De même les assignations V telles que

$$V(1sO - 2sO \leq 0, 1sO < 10, 1sO < 10) \in \{[1,0,1],[0,1,0]\}$$

et celles telles que

$$V(1cDD - 2cDD \leq 0, 1cDD < 5, 2cDD < 5) \in \{[1,0,1],[0,1,0]\}$$

On a donc 6 valuations partielles distinctes pour le triplet $(1sO - 2sO \leq 0, 1sO < 10, 1sO < 10)$ et 6 également pour le triplet $(1cDD - 2cDD \leq 0, 1cDD < 5, 2cDD < 5)$, soit $6 \times 6 = 36$ états différents.

– Π_N a été défini ci-dessus.
– La valuation V_N est donnée, pour tout état $\overline{v}$ de N et pour toute proposition, p de Π_N par : $p \in V(\overline{s})$ ssi $p \in b(\overline{s})$
– On obtient c_N en évaluant les propositions de Π_N dans les ensembles d'états donnés par c_M. Pour tout état $\overline{s} \in N_{ex}$:

$$\begin{aligned} c_M(\overline{s}, \textit{bookshop}) = \quad & \{N_{ex}\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 1sO \geq 10\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 1cDD < 5\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 1cDD \geq 5\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 1sO \geq 10 \wedge 1cDD < 5\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 1sO \geq 10 \wedge 1cDD \geq 5\} \\ c_M(\overline{s}, \textit{sHS}) = \quad & \{N_{ex}\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 2sO < 10\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 2sO \geq 10\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 2cDD \geq 5\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 2sO < 10 \wedge 2cDD \geq 5\} \\ & \cup \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} 2sO \geq 10 \wedge 2cDD \geq 5\} \end{aligned}$$

Pour la coalition binaire $A = \{\textit{bookshop}, \textit{sHS}\}$ on a, modulo l'application des équivalences définissant les variables bSO et $bCDD$:
$c_M(\overline{v}, A) = \{\{N_{ex}\}, \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO < 10 \wedge 1sO - 2sO > 0\}$
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge 1sO - 2sO \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge 1sO - 2sO > 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bCDD < 5 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bCDD \geq 5 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bCDD \geq 5 \wedge 1cDD - 2cDD\, 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO < 10 \wedge bCDD < 5 \wedge 1sO - 2sO > 0 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD < 5 \wedge 1sO - 2sO \leq 0 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD < 5 \wedge 1sO - 2sO > 0 \wedge 1cDD - 2cDD \leq 0\}$,

$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO < 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO \leq 0 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO < 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO \leq 0 \wedge 1cDD - 2cDD > 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO \leq 0 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO > 0 \wedge 1cDD - 2cDD \leq 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO \leq 0 \wedge 1cDD - 2cDD > 0\}$,
$\{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO \geq 10 \wedge bCDD \geq 5 \wedge 1sO - 2sO > 0 \wedge 1cDD - 2cDD > 0\}$

L'examen de la fonction c_N montre que aucun des deux acteurs pris individuellement ne peut assurer la satisfaction à la fois de la condition pour bSO et celle de la condition pour $bCDD$: sHS peut assurer la première mais pas la seconde et $bookshop$ peut assurer la seconde mais pas la première.

En tout état $\overline{v}$ de $\mathfrak{N}$ on a :

$$\mathfrak{N}, \overline{v} \models_{ATL_{\text{KHI}}} \neg\langle\langle bookshop\rangle\rangle \circ bCDD < 5$$

et

$$\mathfrak{N}, \overline{v} \models_{ATL_{\text{KHI}}} \neg\langle\langle sHS\rangle\rangle \circ bSO < 10 \ .$$

Donc

$$\mathfrak{N}, \overline{v} \models_{ATL_{\text{KHI}}} \neg\langle\langle bookshop\rangle\rangle[\![seller]\!]$$

et

$$\mathfrak{N}, \overline{v} \models_{ATL_{\text{KHI}}} \neg\langle\langle sHS\rangle\rangle[\![seller]\!] \ .$$

Par contre en tout état $\overline{v}$ de $\mathfrak{N}$, on a

$$m = \{\overline{s} \mid v \models_{Cond_{\text{KHI}}} bSO < 10 \wedge bCDD < 5 \wedge bSO = 2sO \wedge bCDD = 1cDD\} \in c_N(A, \overline{v}) \ .$$

Et A peut donc assurer la satisfaction de $\circ bSO < 10) \wedge (\circ(bCDD < 5)$ en tout état en jouant le choix m. Donc pour tout état $\overline{v}$ dans $\mathfrak{N}$ on a

$$\mathfrak{N}, \overline{v} \models_{LTL_{\text{KHI}}} (\circ(bSO < 10)) \wedge (\circ(bCDD < 5))$$

et donc

$$\mathfrak{N} \models_{ATL_{\text{KHI}}} \Box((\circ(bSO < 10)) \wedge (\circ(bCDD < 5)))$$

i.e.

$$\mathfrak{N} \models_{ATL_{\text{KHI}}} \langle\langle A\rangle\rangle[\![seller]\!] \ .$$