HAL
open science

# Analyse formelle d'exigences en langue naturelle pour la conception de systèmes cyber-physiques

Aurélien Lamercerie

## To cite this version:

Aurélien Lamercerie. Analyse formelle d'exigences en langue naturelle pour la conception de systèmes cyber-physiques. TALN 2018 - Conférence sur le Traitement Automatique des Langues Naturelles, May 2018, Rennes, France. pp.1-13. hal-01970134

**HAL Id: hal-01970134**

**https://inria.hal.science/hal-01970134v1**

Submitted on 4 Jan 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Analyse formelle d'exigences en langue naturelle pour la conception de systèmes cyber-physiques

Aurélien Lamercerie[1]
(1) Univ Rennes, Inria, IRISA - UMR 6074, F-35000 Rennes, France
aurelien.lamercerie@inria.fr

RÉSUMÉ

Cet article explore la construction de représentations formelles d'énoncés en langue naturelle. Le passage d'un langage naturel à une représentation logique est réalisé avec un formalisme grammatical, reliant l'analyse syntaxique de l'énoncé à une représentation sémantique. Nous ciblons l'aspect comportemental des cahiers des charges pour les systèmes cyber-physiques, c'est-à-dire tout type de systèmes dans lesquels des composants logiciels interagissent étroitement avec un environnement physique. Dans ce cadre, l'enjeu serait d'apporter une aide au concepteur. Il s'agit de permettre de simuler et vérifier, par des méthodes automatiques ou assistées, des cahiers des charges "systèmes" exprimés en langue naturelle. Cet article présente des solutions existantes qui pourraient être combinées en vue de la résolution de la problématique exposée.

ABSTRACT

**Formal analysis of natural language requirements for the design of cyber-physical systems**

This paper focuses on the construction of formal representations of natural language texts. The mapping from a natural language to a logical representation is realized with a grammatical formalism, linking the syntactic analysis of the text to a semantic representation. We target the behavioral aspect of the specifications for cyber-physical systems, *ie* any type of system in which software components interact closely with a physical environment. In this way, the challenge would be to provide assistance to the designer. So, we could simulate and verify, by automatic or assisted methods, "systems" specifications expressed in natural language. This paper presents some existing contributions that could enable progress on this issue.

MOTS-CLÉS : Formalisme grammatical, Représentation sémantique, Grammaire catégorielle, Ingénierie des cahiers des charges, Système cyber-physique, Spécification modale.

KEYWORDS: Grammatical Formalism, Semantic Representation, Categorial Grammar, Requirements Engineering, Cyber-physical System, Modal Specification.

## 1 Introduction

L'objectif de cet article est de proposer les éléments d'une méthodologie permettant de transformer certains fragments de cahiers des charges, exprimés en langue naturelle, en spécifications formelles, vérifiables et exécutables. Notre cadre applicatif porte sur les systèmes cyber-physiques, où des éléments informatiques sont conçus en interaction avec des entités physiques. Nous pouvons donner de nombreux exemples de systèmes cyber-physiques : un avion, une centrale électrique, un réseau ferroviaire, etc.

La conception de ces systèmes implique de décrire de manière précise, cohérente et complète toutes les caractéristiques fonctionnelles et comportementales du système cible. Un cahier des charges comprend généralement un ensemble d'exigences, décrivant précisément toutes les propriétés du système à réaliser. De plus, il rassemble différents points de vues, en mettant notamment en évidence les interactions des entités concernées. C'est une tâche complexe et source d'erreurs. L'outillage informatique qui pourrait être proposé implique le traitement de ces exigences et nécessite préalablement une représentation formelle de ces dernières. Pour l'obtenir, nous pouvons imposer au concepteur des contraintes sur la rédaction du cahier des charges, par exemple en proposant une interface formelle spécifique. Cette orientation représente une difficulté supplémentaire pour le concepteur, qui s'ajoute à celle de la problématique technique auquel il doit répondre. Nous proposons ici une autre approche qui vise à construire automatiquement les représentations formelles à partir d'énoncés exprimés en langue naturelle.

Nous nous intéressons plus particulièrement à l'aspect comportemental des systèmes étudiés, spécifié avec un ensemble d'exigences. Celles-ci décrivent des enchaînements d'action, et contiennent généralement des contraintes temporelles et des modalités déontiques. Les contraintes temporelles précisent la portée d'un évènement ou d'une propriété, tandis que les modalités déontiques expriment des obligations, des interdictions ou des possibilités. Par exemple, les énoncés suivants caractérisent l'état attendu d'une porte d'un système quelconque :

— La porte doit s'ouvrir lorsqu'un ticket est présenté. (obligation)
— L'ouverture de la porte est interdite après 21 heures. (interdiction)
— La porte peut s'ouvrir manuellement. (possibilité)

La logique de notre démarche est d'aboutir à la proposition de logiciels intégrés aux outils existants, favorisant ainsi une bonne acceptation des utilisateurs. Cet objectif tend à exclure l'utilisation d'un langage de domaine spécifique (DSL) pour privilégier la langue naturelle.

En traitement automatique des langues, les approches à dominante statistique sont généralement distinguées de celles à dominante symbolique. Nous proposons d'utiliser un formalisme de haut-niveau, à dominante symbolique : les grammaires catégorielles (Steedman, 2000; Moot & Retoré, 2012). Notre volonté est de proposer une représentation la plus juste et fine possible des exigences exprimées. Les grammaires catégorielles permettent une analyse compositionnelle, un développement incrémental et une interface aisée avec la sémantique. Ces points sont pertinents et importants si nous voulons obtenir une représentation qui capture le sens logique des phrases, avec un résultat manipulable et exploitable pour la vérification d'exigences.

Un cahier des charges peut se ramener à un ensemble de propriétés se rapportant aux composants d'une architecture. Partant de cet ensemble, nous voulons aboutir à une représentation du système. Sur le plan opératoire, nous pouvons décomposer notre problématique en deux phases (figure 1).

La première phase est l'analyse formelle des exigences, qui doit permettre d'aboutir à leur représentation sémantique. Une exigence est un énoncé en une phrase, parfois quelques phrases, donnant une spécification comportementale sur un point précis. La phrase "Si un ticket est inséré, la porte peut s'ouvrir" constitue un exemple d'exigence sur une porte dont l'ouverture est conditionnelle. Nous proposons de construire les représentations sémantiques en nous appuyant sur un formalisme grammatical.

La seconde phase consiste en la composition de ces exigences pour obtenir une représentation du comportement du système. Deux exigences ne se composeront pas nécessairement de la même manière ; il est donc nécessaire que les propriétés algébriques du modèle utilisé permettent l'usage de

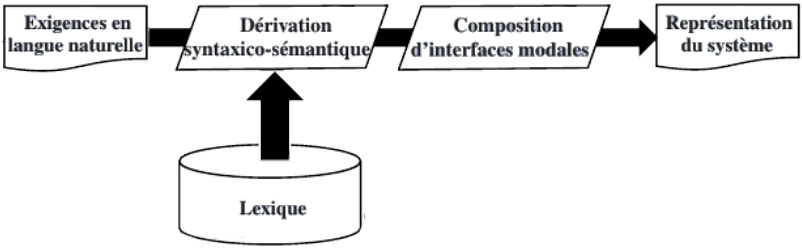


FIGURE 1 – Vue d'ensemble de la méthodologie proposée

différentes opérations de composition.

Cet article est un document d'analyse et de positionnement, présentant différents concepts qui pourraient être combinées. La section 2 introduit un formalisme pour l'analyse sémantique d'une exigence, les grammaires catégorielles combinatoires (Steedman, 2000), tandis que la représentation du comportement des systèmes techniques est étudiée, section 3, avec l'introduction des spécifications modales (Larsen & Thomsen, 1988; Raclet, 2007). Quelques travaux préliminaires sont présentés dans la section 4. Les perspectives offertes par l'approche proposée sont finalement discutées dans la section 5.

# 2 Analyse d'exigence

L'approche que nous proposons vise à construire une représentation logique d'un énoncé en "capturant" finement sa sémantique. Suivant les travaux de Montague (Montague, 1974), nous considérons le sens d'une phrase comme lié à sa construction syntaxique. Nous partons d'une décomposition de l'énoncé à l'aide d'un formalisme grammatical, pour ainsi définir le comportement syntaxique de l'énoncé cible. L'idée est ensuite de relier cette décomposition à une représentation formelle.

Comme indiqué en introduction, nous proposons d'utiliser les grammaires catégorielles, un formalisme de haut-niveau pour l'analyse de textes. Les grammaires catégorielles ont pris forme dans les années 50 et 60, à partir des travaux de Adjukiewicz (Ajdukiewicz, 1935), Bar-Hillel (Bar-Hillel, 1953) et Lambek (Lambek, 1958). Elles reposent sur la notion de catégories primitives et de catégories dérivées (on parle aussi de types). Elles offrent la possibilité de décomposer une phrase en catégories, les catégories composées mettant en évidence les liens syntaxiques du langage ciblé. Les grammaires catégorielles intègrent également des règles de dérivation qui permettent de combiner les éléments d'une phrase.

Du point de vue de la théorie grammaticale, il est important de considérer l'expressivité des formalismes étudiés. Pour les grammaires catégorielles classiques, elle est équivalente aux grammaires non-contextuelles (Bar-Hillel et al, 1963). Plusieurs extensions ont donc été proposées pour dépasser cette contrainte, dont la grammaire catégorielle combinatoire (CCG), introduite par Mark Steedman en 1989 (Steedman, 2000). Ce formalisme utilise un ensemble d'opérateurs pour saisir d'autres aspects linguistiques. Dans cette grammaire, la sémantique des mots est guidée tout au long de l'arbre de dérivation en combinant les sémantiques des autres mots jusqu'à produire la sémantique complète de la phrase, à la Montague (Montague, 1974). Elle est particulièrement intéressante dans notre cas

pour ses liens avec le lambda-calcul, qui pourrait être une représentation formelle intermédiaire avant d'aboutir à d'autres représentations logiques.

Concrètement, les grammaires CCG permettent de relier la syntaxe des énoncés étudiés à leur représentation sémantique sous-jacente. Elles se définissent avec un ensemble de catégories basiques, un ensemble de catégories dérivées, des règles de calcul syntaxiques et un lexique de mots.

Les catégories primitives correspondent aux catégories syntaxiques habituellement utilisées dans les grammaires traditionnelles. Cet ensemble de catégories primitives n'est pas universel mais fait partie de la définition de la grammaire. Il est possible de se restreindre à deux catégories : *S* pour la phrase ("sentence" en anglais), *N* pour les noms communs et les noms propres. Il est également possible d'étendre cet ensemble en ajoutant des catégories, par exemple pour les adjectifs ou les pronoms. De même, la grammaire peut-être raffiner avec des sous-catégories, comme sur la figure 2.

Les catégories dérivées sont construites à partir des catégories primitives. Elles peuvent se définir récursivement avec les règles syntaxiques : soit $X$ et $Y$ deux catégories, alors $X/Y$ et $X\backslash Y$ sont également des catégories [1].

Les règles syntaxiques permettent de produire l'analyse sémantique en combinant les entrées lexicales. Les grammaires catégorielles classiques proposent deux règles de calcul, également présentes avec les CCG : l'application en avant ( $>$ ) et l'application en arrière ( $<$ ). Chacune de ces règles permet de former une nouvelle catégorie en combinant une catégorie de type $X/Y$ ou $X\backslash Y$ avec une catégorie de type $Y$. Formellement, elles peuvent se définir comme suit :

— $X/Y, Y \rightarrow X\ (>)$
— $Y, X\backslash Y \rightarrow X\ (<)$

Les CCG enrichissent les grammaires catégorielles classiques avec de nouvelles règles de calcul. Nous définissons ici les règles de composition harmoniques, en avant ( $B>$ ) et en arrière ( $B<$ ) :

— $X/Y, Y/Z \rightarrow X/Z\ (B>)$
— $Y\backslash X, Z\backslash Y \rightarrow Z\backslash X\ (B<)$

D'autres règles sont également proposées : des règles de compositions croisées, des règles de composition mixtes et des règles de changement de types. L'ensemble de ces règles offre différentes façons de combiner les catégories, et introduisent un certain degré d'associativité. Elle permettent ainsi de traiter des aspects linguistiques plus riches.

Finalement, le lexique permet de définir la grammaire qui correspond à un langage donné. Il est constitué d'un ensemble d'entrées lexicales associant un mot et un ensemble de types, auxquelles il est possible d'adjoindre une représentation sémantique. Cette dernière peut être donnée sous la forme d'expressions en lambda-calcul [2]. Par exemple, le mot "gate" pourra être associé à la catégorie N, tandis que la modalité "may" serait associée à la catégorie $(S\backslash NP)/(S\backslash NP)$. Ces deux entrées

1. Il existe plusieurs systèmes de notation. Dans la notation de Lambek, $X/Y$ se lit $X$ sur $Y$ et signifie qu'un mot (ou groupe de mots) de catégorie $Y$ est attendu à droite pour former la catégorie $X$. Par symétrie, $X\backslash Y$ se lit $X$ sous $Y$ et signifie qu'un mot ou groupe de mots de catégorie $X$ est attendu à gauche pour former la catégorie $Y$. Dans la notation de Steedman, $X/Y$ et $X\backslash Y$ définissent une catégorie avec un argument de type Y et un résultat de type X. Une barre oblique (/) indique que l'argument doit apparaître à droite, une barre oblique inversée (\) indique que l'argument doit apparaître à gauche. C'est cette dernière notation que nous utilisons dans cet article.

2. Le lambda-calcul est un formalisme qui permet de définir et caractériser les fonctions mathématiques. En particulier, ce formalisme offre la possibilité de modéliser des expressions fonctionnelles et leur évaluation en manipulant des expressions, appelées lambda-termes, dans lesquelles une abstraction désigne une définition de fonction. Ce formalisme a été proposé par Alonzo Church dans les années 1930.

pourraient être rattachées respectivement aux formules $gate$ et "$\lambda x.(x, may)$".

La figure 2 montre un exemple d'analyse avec le parseur EasyCCG [3] (Lewis & Steedman, 2014) pour la phrase "After a ticket is inserted, the gate may open". L'analyse syntaxique doit permettre de réduire la séquence des catégories associées aux mots à la catégorie principale de la phrase S, ce que nous observons sur cette exemple. Les éléments entre crochets correspondent à des étiquettes grammaticales permettant d'affiner l'analyse syntaxique.

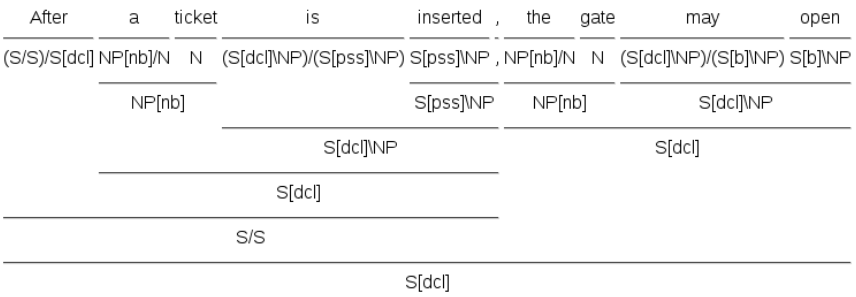


FIGURE 2 – Dérivation CCG de la phrase "After a ticket is inserted, the gate may open" construite avec le parser EasyCCG

Les dérivations obtenues avec une analyse CCG peuvent être reliées à une représentation sémantique, en s'appuyant sur les associations définies dans le lexique. Des algorithmes d'analyse CCG en temps polynomial existent (Vijay-Shanker & Weir, 1993). La mise en oeuvre de ce modèle s'est traduite dans le développement de plusieurs outils, dont l'outil Boxer/C&C (Bos, 2015) utilisé dans les travaux préliminaires présentés en section 4.

# 3 Représentation du comportement d'un système

Les systèmes cyber-physiques sont généralement complexes. Ils résultent de l'assemblage de plusieurs composants, conçus par des équipes travaillant indépendamment. Ces équipes s'accordent sur les attentes pour chaque composant et leur intégration commune en s'appuyant sur les spécifications techniques issues du cahier des charges. Pour vérifier la consistance et la complétude des exigences d'un cahier des charges, il est nécessaire de disposer d'un formalisme de spécification. Ce formalisme doit offrir un bon compromis en termes d'expressivité, de propriétés algébriques et de complexité algorithmique.

Les automates ne permettent pas d'exprimer la variabilité des points de vues et des composants. Les automates d'interfaces (de Alfaro & Henzinger, 2001) n'ont pas l'opération de conjonction, tandis que les logiques temporelles LTL et CTL (Clarke & Emerson, 1981; Clarke *et al.*, 1986) n'ont pas la composition parallèle. Le $\mu$-calcul modal (Arnold & Niwiński, 2001; Arnold *et al.*, 2003) possède les bonnes propriétés algébriques, avec un niveau d'expressivité élevé mais une complexité de calcul également importante ($O(2^{2^N})$).

3. EasyCCG est un parseur CCG open-source. Il est disponible à l'adresse suivante : http ://home-pages.inf.ed.ac.uk/s1049478/easyccg.html.

Les spécifications modales (Raclet, 2007) représentent le compromis recherché. En effet, correspondant aux systèmes de transitions modaux déterministes (Larsen & Thomsen, 1988), ce formalisme intègre les opérations de conjonction, de composition et de quotient avec une complexité algorithmique polynomiale. Il est donc adapté pour la représentation d'interfaces des différents composants d'un système. Certes, les spécifications modales sont strictement moins expressives que le $\mu$-calcul modal . Cette limitation est néanmoins acceptable pour formaliser les propriétés de sureté des cahiers des charges de systèmes cyber-physiques.

Ce formalisme permet de représenter des automates dont les transitions sont typées avec les modalités *must* et *may*, permettant de définir un ensemble de modèles. De manière informelle, une transition de type *must* ( *obligatoire*) doit être présente dans chaque composant qui implémente la spécification modale, alors qu'une transition de type *may* ( *facultative*) peut ne pas l'être.

Les spécifications modales sont des systèmes de transitions modaux déterministes (Raclet, 2007). Les systèmes de transitions modaux ont été introduits par Larsen et Thomsen dans les années 80 (Larsen & Thomsen, 1988; Jonsson & Larsen, 1991) pour étudier le raffinement d'actions, opération qui consiste à remplacer des transitions pour relier les descriptions d'un système à différents niveaux d'abstraction.

Nous reprenons la définition formelle de J.-B. Raclet (Raclet, 2007). Une spécification modale est un tuple $S = (Q, q_0, \Sigma, \Delta^m, \Delta^M)$. $Q$ est un ensemble fini d'états. $q_0$ est l'unique état initial. $\Sigma$ est un ensemble fini d'actions. $\Delta^m \subseteq Q \times \Sigma \times Q$ est l'ensemble déterministe des transitions facultatives ; $\Delta^M \subseteq Q \times \Sigma \times Q$ est l'ensemble déterministe des transitions obligatoires. Enfin, la condition de consistance suivante doit être respectée : $\Delta^M \subseteq \Delta^m$ et $q_0 \in Q$. Cette condition signifie que chaque transition obligatoire (appartenant à l'ensemble $\Delta^M$) est également une transition facultative (appartenant à l'ensemble $\Delta^m$).

Le modèle d'une spécification modale est défini à partir de la relation de satisfaction, notée $\models$. Soit $S = (Q, q_0, \Sigma, \Delta^m, \Delta^M)$ une spécification modale. Soit $M = (R, r_0, \Sigma, \Gamma \subseteq R \times \Sigma \times R)$ avec $\Gamma$ déterministe. $M$ est un modèle de $S$, noté $M \models S$, si, et seulement si, il existe $\rho \subseteq R \times Q$ tel que :

1. $(r_0, q_0) \in \rho$
2. $\forall (r, q) \in \rho, \forall \sigma \in \Sigma$
   (a) $\forall r' \in R, (r, \sigma, r') \in \Gamma \Rightarrow \exists q' \in Q$ tel que $(q, \sigma, q') \in \Delta^m$ et $(r', q') \in \rho$
   (b) $\forall q' \in Q, (q, \sigma, q') \in \Delta^M \Rightarrow \exists r' \in R$ tel que $(r, \sigma, r') \in \Gamma$

Il est également possible de définir une représentation graphique de spécifications modales. Ces représentations produisent des exemples visuels sous la forme de graphes avec des arcs continus ou pointillés. La présence d'un arc continu signifie que la transition est obligatoire ("must"), la présence d'un arc en pointillé que la transition est facultative ("may") et l'absence d'arc indique que la transition est interdite dans toute réalisation.

La figure 3 montre une spécification modale (à gauche) et deux modèles possibles de cette spécification (à droite), où $a$ et $b$ sont deux actions (ou évènements).

Les spécifications modales sont dotées de propriétés algébriques particulièrement intéressantes. Ces propriétés sont détaillées dans (Raclet *et al.*, 2010). On y trouvera en particulier les définitions formelles (et preuves associées) pour la composition ($S_1 \otimes S_2$), la conjonction ($S_1 \wedge S_2$) et le quotient ($S_1/S_2$) de spécifications modales (avec $S_1$ et $S_2$ deux spécifications modales).

Nous précisons ici quelques points importants : la notion de raffinement, et les caractéristiques algébriques des opérations évoquées. Une spécification modale $S_1$ est le raffinement d'une spéci-

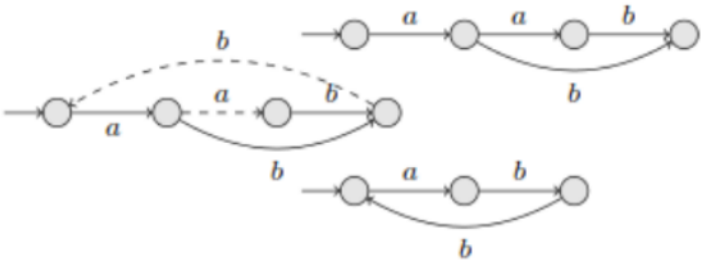


FIGURE 3 – Une spécification modale et deux modèles possibles

fication $S_2$ si, et seulement si, les modèles de $S_1$ sont aussi des modèles de $S_2$. Formellement, le raffinement, noté $\preceq$, se définit comme suit : $S_1 \preceq S_2$ si, et seulement si, pour tout modèle $M$, nous avons $M \models S_1 \Rightarrow M \models S_2$. Nous pouvons dès lors donner les caractéristiques algébriques des opérations de composition, conjonction et quotient, en nous appuyant sur le raffinement :

— $S_1 \otimes S_2 = min\{S \mid \forall M_1 \models S_1, \forall M_2 \models S_2, M_1 \times M_2 \models S\}$
— $S_1 \wedge S_2 = min\{S \mid \forall M, M \models S_1 et M \models S_2 \Rightarrow M \models S\}$
— $S_1 / S_2 = max\{S \mid S_2 \otimes S \leq S_1\}$

Les spécifications modales apparaissent comme un formalisme pertinent pour la représentation des exigences d'un système réactif. Elles peuvent prendre la forme d'automates qu'il est possible d'exécuter pour valider un cahier des charges. De plus, elles sont implémentables avec une complexité raisonnable, et permettent de représenter les modalités déontiques et temporelles présentes dans la définition d'un système.

# 4 Travaux préliminaires

Nous proposons de montrer une mise en oeuvre possible des concepts exposés dans les sections précédentes sur un cas d'étude. L'objectif visé est la construction de la spécification modale d'un système dont le comportement est décrit en langue naturelle. Pour atteindre ce but, nous présentons une méthode de traitement sur un ensemble d'exigences. Celle-ci doit permettre de relier les dérivations syntaxiques de ces exigences à des représentations sémantiques pertinentes.

L'exemple présenté ici repose sur des expérimentations préliminaires (Boudaoud, 2016) portant sur un système représentant un garage de voitures (Benveniste *et al.*, 2018). La figure 4 propose une vue d'ensemble du document de spécifications, composé de plusieurs sections (*gate*, *payment*, *supervisor*).

Pour ces travaux préliminaires, les données utilisées proviennent d'un cas d'étude théorique. De plus, les exigences exprimées en plusieurs phrases, avec des formulations imposant la prise en compte du contexte, ont été mises de côté. Chaque exigence est donc définie en une phrase et sans pronom,

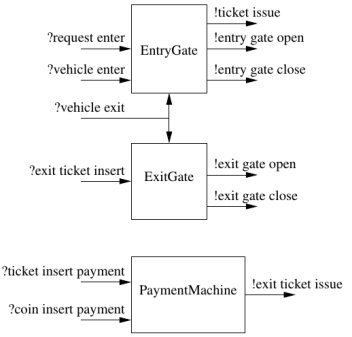


FIGURE 4 – Architecture du système "The Parking Garage" (Benveniste *et al.*, 2018)

chaque entité devant être explicitement désignée dans l'exigence [4].

Les représentations sémantiques sont obtenues à partir de dérivations CCG, comme définies section 2. Concrètement, chaque élément lexical est associé à une catégorie lexicale et une représentation sémantique exprimée en lambda-calcul. Il est également possible de donner plusieurs représentations pour un même élément lexical (dans le cas où l'élément lexical serait associé à plusieurs catégories syntaxiques). Le traitement implique ensuite de déterminer les fonctions et les arguments, et d'appliquer récursivement les fonctions sur les arguments jusqu'à obtenir la représentation sémantique complète de l'exigence ciblée.

Le passage des formules en lambda-calcul vers les spécifications modales comporte plusieurs difficultés, dont la mise en relation des termes de l'énoncé (source) aux états et actions de la spécification (cible). Au niveau de l'implémentation, un module spécifique pourrait permettre de gérer cette transformation. Ce module doit également intégrer la résolution des références aux concepts, par exemple en s'appuyant sur une ontologie du domaine d'étude.

Un automate peut être spécifié sous la forme d'une expression régulière, dont la définition peut prendre en compte les opérateurs algébriques tels que la concaténation, le choix, la répétition ou le mot vide. Des règles de transformation associant une expression régulière à chaque terme en langue naturelle peuvent être proposées en s'appuyant sur cette définition. L'application de ces règles permet de décrire les propriétés d'états en expression régulière. Sur le même principe, nous pouvons associer des transitions aux termes en langue naturelle définissant des modalités. La combinaison de ces relations permet de construire la représentation des états et transitions sous la forme d'une spécification modale, à partir de l'arbre de dérivation CCG.

Ce cas d'étude a été étudié en détail, du point de vue *"conception de système"*, dans (Benveniste *et al.*, 2018). Il a été mis en oeuvre avec l'outil MICA (Caillaud, 2011), une bibliothèque qui permet de définir des systèmes complexes en utilisant une syntaxe du langage O'Caml. Dans cette mise en oeuvre, la formalisation des exigences a été réalisée manuellement (les exigences ont été traduites dans la syntaxe exigée par l'outil Mica). L'objectif serait d'enrichir cet outil pour pouvoir traiter

4. Par exemple, l'exigence "Quand la porte est fermée, elle ne peut pas être ouverte tant qu'aucun ticket n'est inséré." n'a pas été traitée sous cette forme car la prise en compte du pronom "elle" pose quelques difficultés. Elle peut l'être en revanche sous la forme suivante : ""Quand la porte est fermée, la porte ne peut pas être ouverte tant qu'aucun ticket n'est inséré". Néanmoins, des travaux complémentaires, en s'appuyant sur les DRT, devraient permettre de lever cette restriction.

directement les exigences exprimées en langue naturelle. Dans le tableau 1, nous reprenons quelques exigences portant sur la section "gate".

| *Id.* | *Exigence* |
|---|---|
| 1 | Entry gate must open once a ticket has been issued. |
| 2 | When a ticket has been issued, entry gate must open. |
| 3 | When a ticket has been issued, entry gate opening is obligatory. |
| 4 | Afer an exit ticket is inserted, exit gate must open. |

TABLE 1 – Quelques exigences du système "The parking garage"

Il est à noter que les exigences contiennent plusieurs caractéristiques linguistiques qui doivent être prises en compte. Elles peuvent décrire des enchaînements d'évènements, avec des modalités temporelles. Elles peuvent également contenir des éléments qui font référence à des entités de l'environnement du système (par exemple un véhicule), ou des informations implicites. Les spécifications contiennent aussi des modalités d'obligation et d'interdiction. L'extraction des informations nécessaires pour le passage vers une spécification modale implique une bonne évaluation de ces différents aspects.

L'outil Boxer, développé par Johan Bos (Bos, 2015), a été utilisé pour traiter les exigences et construire des représentations sémantiques. Boxer/C&C est un analyseur sémantique de textes en anglais. Il s'appuie sur la grammaire catégorielle combinatoire et la théorie de représentation du discours (Kamp, 1981), et permet de construire des représentations sémantiques sous différentes formes. La figure 5 donne une représentation de l'exigence 4 du tableau 1.

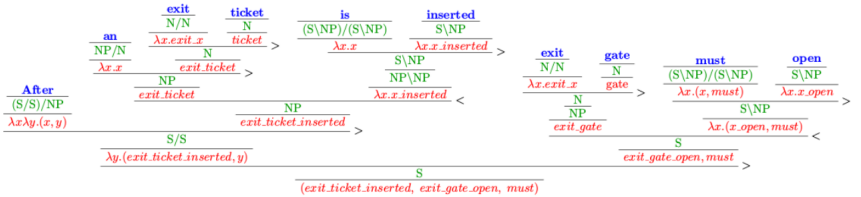


FIGURE 5 – Représentation graphique de l'analyse sémantique pour l'exigence 4

Le lien entre Boxer et Mica n'a pas encore été réalisé. Sur le principe, il s'agit de générer du code O'Caml à partir des représentations formelles obtenues. La figure 6 donne une vue du résultat que nous pourrions obtenir sous la forme d'une spécification modale en prenant en compte les exigences de la section "Gate". Cette représentation intègre l'analyse de l'exigence 4 (figure 5). On retrouve cette propriété au niveau des états 0 et 1.

Même s'il est incomplet, le résultat obtenu montre la pertinence de l'approche. Il serait souhaitable de développer un outil complet permettant l'implémentation d'interfaces modales pour des composants spécifiés en langue naturelle. Sa réalisation pourrait s'appuyer sur les outils présentés, Boxer pour la dérivation CCG, et Mica pour les interfaces modales, ou d'autres outils aux caractéristiques similaires. Pour valider cette démarche, il conviendrait également d'évaluer l'approche sur l'ensemble des exigences du cas d'étude, ainsi que sur des données issues d'un cas réel.

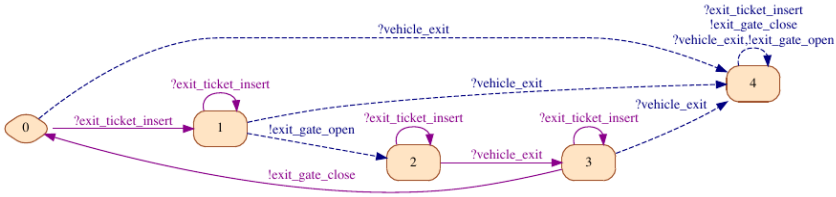


FIGURE 6 – Représentation graphique de la spécification modale pour la section "Gate" du cas d'étude "The Parking garage" (Benveniste *et al.*, 2018)

# 5 Discussion et perspectives

L'analyse des besoins est la première étape du cycle de développement d'un système. L'élaboration du cahier des charges intervient donc très tôt dans ce cycle ; le coût d'une erreur non détectée à ce stade est important. Le passage des spécifications en langage naturel vers des spécifications formelles vérifiables suscite un intérêt évident. Plusieurs approches ont déjà été proposées. Elles utilisent généralement une représentation intermédiaire pour la formalisation de spécifications en langue naturelle, par exemple une modélisation des connaissances par une ontologie (Sadoun *et al.*, 2013), des patrons de spécifications de propriétés temporelles (Dwyer *et al.*, 1999; Konrad & Cheng, 2005), un langage naturel contrôlé (Fuchs & Schwitter, 1995).

A contrario, l'approche que nous proposons vise à construire automatiquement les représentations formelles à partir d'énoncés exprimés en langue naturelle. Même si l'écriture des énoncés devra être d'abord restreinte à un langage contrôlé, notre objectif est de permettre de dépasser cette contrainte et rejoindre dans une certaine mesure la flexibilité du langage naturel.

Pour atteindre notre objectif, plusieurs étapes sont nécessaires. La première étape consiste en la dérivation d'une analyse syntaxique de l'énoncé, pour aboutir à un arbre syntaxique. La deuxième étape vise à relier l'analyse syntaxique à une représentation formelle intermédiaire, par exemple le lambda-calcul. Nous pensons possible de passer de cette représentation à une autre forme. Nous avons notamment montré l'intérêt des spécifications modales pour la vérification d'un cahier des charges.

Concernant le passage de la langue naturelle vers une représentation formelle, nous avons évoqué une première piste, à savoir l'utilisation des CCG, couplée aux DRT si nous voulons traiter les pronoms. D'autres options sont également envisageables et pourraient être étudiées. Nous pensons notamment aux travaux autour du "Grammatical Framework" [5], également fondé sur les grammaires catégorielles et la théorie des types. Nous songeons aussi aux Grammaires Catégorielles Abstraites (de Groote, 2001), formalisme qui généralise les grammaires catégorielles en traitant directement les catégories et le lambda-calcul.

Nous avons évoqué la théorie de représentation du discours (DRT), proposé par Hans Kamp en 1983 (Kamp, 1981). Ce modèle permet de construire une représentation d'un texte en examinant le contenu sémantique dépendant du contexte. Il permet ainsi de traiter certaines difficultés linguistiques, en particulier les mots grammaticaux se substituant à un élément donné (pronoms). A la base de la DRT, il y a l'idée que l'interprétation d'un discours s'appuie sur une représentation sémantique, chaque

5. Le système GF, qui combine langage source et langage cible, a été utilisé dans différents projets. Il est accessible à l'adresse suivante : https ://www.grammaticalframework.org/

phrase enrichissant cette représentation. Cette représentation est appelée "structure de représentation du discours" (DRS). La DRT intègre également une procédure de construction, pour mettre à jour la représentation à chaque nouvelle phrase, et un ensemble de modèle sémantique, pour évaluer les DRS. Ce formalisme pourrait être exploité conjointement aux CCG pour augmenter la couverture des phénomènes linguistiques.

La validation de la méthodologie proposée passe par une mise en oeuvre concrète. Une première étape serait de développer un prototype visant l'implémentation de spécifications modales à partir d'exigences exprimées dans un langage contraint, le plus proche possible de la langue naturelle. Dans un second temps, il s'agirait de lever les contraintes sur le texte pour se rapprocher au maximum de la flexibilité de la langue naturelle.

De nombreuses questions se posent alors, en particulier pour le traitement des énoncés ambigus. L'interaction entre le programme et l'utilisateur est une première réponse. Nous pourrions imaginer que le système propose plusieurs variantes pour l'énoncé. Cependant, cela demanderait un pré-requis important pour l'usage de l'outil, à savoir la maîtrise du langage formel utilisé. Cette solution ne serait pas satisfaisante sous cette forme. Il serait donc intéressant de travailler sur le cheminement inverse, c'est-à-dire le passage d'une représentation formelle vers un énoncé en langue naturelle intelligible.

Nous pourrions d'ailleurs pousser la réflexion sur le développement de l'interaction. La théorie de la représentation du discours pourrait être une source d'inspiration. L'idée serait la suivante : l'analyse de l'énoncé permet de construire une première représentation avec un certain niveau d'ambiguïté ; les interactions du système avec l'utilisateur conduisent à la mise à jour de cette représentation, qui est validée lorsqu'il n'y a plus d'ambiguïté.

L'utilisation d'une ontologie pour représenter les concepts du domaine est une autre piste, alternative ou complémentaire. Celle-ci nous semble particulièrement adaptée pour le traitement des ambiguïtés dans les groupes nominaux. En effet, un groupe nominal, éventuellement complexe, doit correspondre à un concept de l'ontologie, et ainsi être vu comme atomique. Les synonymes pourraient être traités de manière similaire, un même concept pouvant correspondre à plusieurs groupes nominaux.

Ces deux axes pourraient être complémentaires. La construction d'une ontologie est cependant une tâche délicate qui demande un investissement important. Travailler sur une ontologie incomplète pourrait être une nécessité. Il serait instructif d'examiner les possibilités d'apprentissage des concepts par le système, et de la mise à jour interactive de l'ontologie de référence. Par exemple, le cahier des charges pourrait contenir des définitions qui pourraient être prises en compte pour mettre à jour l'ontologie. De même, les différents échanges du système avec l'utilisateur pourraient être pris en compte, avec un certain degré de confiance sur les concepts enregistrés. Par ricochet, de nouvelles problématiques apparaissent : comment quantifier ce degré de confiance ?

Et même au delà, l'un des objectifs étant de vérifier la cohérence des exigences, une réflexion devra être menée sur la mise en évidence des incohérences. Autrement dit, quel résultat proposer lorsque le cahier des charges est inconsistant ? Il pourrait alors être utile d'offrir au concepteur la possibilité de simuler un cahier des charges, traduit dans un formalisme tel que les spécifications modales, et de modifier les spécifications de manière interactive, en cours de simulation.

Un autre axe de travail consisterait à étendre les interfaces modales à des formalismes plus expressifs, et à tenter de prendre en compte les propriétés numériques des systèmes cyber-physiques. Les propriétés numériques permettent de définir des enveloppes de trajectoires pour les systèmes cyber-physiques, prenant généralement la forme d'équations ou d'inclusions différentielles qui caractérisent la dynamique en temps continu du système.

Dans cet article, nous avons mis en lumière différents formalismes qui nous semblent utiles pour répondre à notre problématique. Notre présentation est loin d'être exhaustive. Il existe en effet de nombreux formalismes grammaticaux, sans réel consensus. Il serait intéressant de comparer ces formalismes, ce qui pourrait nous conduire à orienter nos travaux dans de nouvelles directions.

# Références


AJDUKIEWICZ K. (1935). Die syntaktische konnexität. *Studia Philosophica*, **1**, 1–27. Traduction française de (Gan-Krzywoszyńska, 2007).

ARNOLD A. & NIWIŃSKI D. (2001). *Rudiments of [mu]-calculus*. Studies in Logic and the Found. Elsevier.

ARNOLD A., VINCENT A. & WALUKIEWICZ I. (2003). Games for synthesis of controllers with partial observation. *Theoretical Computer Science*, **303**(1), 7–34. Logic and Complexity in Computer Science.

BAR-HILLEL Y. (1953). A quasi-arithmetical notation for syntactic description. *Language*, **29**, 47–58.

BENVENISTE A., CAILLAUD B., NICKOVIC D., PASSERONE R., RACLET J.-B., REINKEMEIER P., SANGIOVANNI-VINCENTELLI A., DAMM W., HENZINGER T. A. & LARSEN K. G. (2018). Contracts for system design. *Foundations and Trends® in Electronic Design Automation*, **12**(2-3), 124–400.

BOS J. (2015). Open-domain semantic parsing with boxer. *Proceedings of the 20th Nordic Conference of Computational Linguistics*, p. 301–304.

BOUDAOUD S. R. (2016). Spécification d'exigences en langue naturelle, avec automates et logique. Master's thesis, Université de Rennes 1. Private communication.

CAILLAUD B. (2011). Mica : A Modal Interface Compositional Analysis Library.

CLARKE E. M. & EMERSON E. A. (1981). Synthesis of synchronization skeletons for branching time temporal logic. In *Logic of programs : Workshop*, volume 131, p. 244–263.

CLARKE E. M., EMERSON E. A. & SISTLA A. P. (1986). Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.*, **8**(2), 244–263.

DE ALFARO L. & HENZINGER T. A. (2001). Interface automata. *SIGSOFT Softw. Eng. Notes*, **26**(5), 109–120.

DE GROOTE P. (2001). Towards abstract categorial grammars. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, p. 252–259 : Association for Computational Linguistics.

DWYER M. B., AVRUNIN G. S. & CORBETT J. C. (1999). Patterns in property specifications for finite-state verification. In *Proceedings of the 21st International Conference on Software Engineering*, ICSE '99, p. 411–420, New York, NY, USA : ACM.

FUCHS N. E. & SCHWITTER R. (1995). Specifying logic programs in controlled natural language. *CoRR*, **abs/cmp-lg/9507009**.

GAN-KRZYWOSZYŃSKA K. (2007). La connexion syntaxique. *Philosophia Scientiæ*, **11-2**, 97–120.

JONSSON B. & LARSEN K. G. (1991). Specification and refinement of probabilistic processes. In *[1991] Proceedings Sixth Annual IEEE Symposium on Logic in Computer Science*, p. 266–277.

KAMP H. (1981). A theory of truth and semantic representation. In J. A. G. GROENENDIJK, T. M. V. JANSSEN & M. B. J. STOKHOF, Eds., *Formal Methods in the Study of Language*, volume 1, p. 277–322. Amsterdam : Mathematisch Centrum.

KONRAD S. & CHENG B. H. C. (2005). Real-time specification patterns. In *Proceedings of the 27th International Conference on Software Engineering*, ICSE '05, p. 372–381, New York, NY, USA : ACM.

LAMBEK J. (1958). The mathematics of sentence structure. *American Mathematical Monthly*, **154-170**.

LARSEN K. G. & THOMSEN B. (1988). A modal process logic. In *Proceedings. Third Annual Symposium on Logic in Computer Science*, p. 203–210.

LEWIS M. & STEEDMAN M. (2014). A* ccg parsing with a supertag-factored model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processin*.

MONTAGUE R. (1974). *Formal Philosophy : Selected Papers of Richard Montague*. A Yale Paperbound. Yale University Press.

MOOT R. & RETORÉ C. (2012). *The Logic of Categorial Grammars : A Deductive Account of Natural Language Syntax and Semantics*. FoLLI-LNCS. Springer.

RACLET J. (2007). *Quotient de spécifications pour la réutilisation de composants*. PhD thesis, Université de Rennes 1.

RACLET J., BADOUEL E., BENVENISTE A., CAILLAUD B., LEGAY A. & PASSERONE R. (2010). A modal interface theory for component-based design. *Fundamenta Informaticae*, **108**(1-2), 119–149.

SADOUN D., DUBOIS C., GHAMRI-DOUDANE Y. & GRAU B. (2013). From natural language requirements to formal specification using an ontology. In *IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI 2013)*, p. 755–760, Herndon, VA, United States.

STEEDMAN M. (2000). *The syntactic process*. MIT Press.

VIJAY-SHANKER K. & WEIR D. J. (1993). Parsing some constrained grammar formalisms. *Comput. Linguist.*, **19**(4), 591–636.