# Evaluating the Assignment of Behavioral Goals to Coalitions of Agents

Christophe Chareton, Julien Brunel, David Chemouil

# Evaluating the Assignment of Behavioral Goals to Coalitions of Agents

Christophe Chareton[1], Julien Brunel[2], David Chemouil[2]

[1] École Polytechnique de Montréal, Montreal (Quebec), Canada
[2] Onera/DTIM, Toulouse, France

**Abstract.** We present a formal framework for solving what we call the "assignment problem": given a set of behavioral goals for a system and a set of agents described by their capabilities to make the system evolve, the problem is to find a "good" assignment of goals to (coalitions of) agents. To do so, we define KORE, a core modelling framework as well as its semantics in terms of a strategy logic called USL. In KORE, agents are defined by their capabilities, which are pre- and post-conditions on the system variables, and goals are defined in terms of temporal logic formulas. Then, an assignment associates each goal with the coalition of agents that is responsible for its satisfaction. Our problem consists in defining and checking the *correctness* of this assignment. We define different criteria for modelling and formalizing this notion of correctness. They reduce to the satisfaction of USL formulas in a structure derived from the capabilities of agents. Thus, we end up with a procedure for deciding the correctness of the assignment. We illustrate our approach using a toy example featuring exchanges of resources between a provider and two clients.

## 1 Introduction

The question of assigning behavioral goals to coalitions of agents (*i.e.* sets of active entities), the capabilities of whom are known, is a fundamental and recurring problem in various areas of software and systems engineering. We call it the *assignment problem*. In this paper, we propose a formalization of this problem and then describe various criteria to assess an assignment formally.

First, let us illustrate various situations where this problem arises; this will not only demonstrate the ubiquity of this problem but also enable us to delineate the most salient aspects that ought to be addressed in the formalization.

In the field of Requirements Engineering (RE), for instance, several modeling languages have been proposed that each partly feature the concepts just mentioned. Thus, KAOS [Let02, LVL02b, vL09, vL03], a so-called *goal-oriented* modeling language, features *behavioral* goals that may be formalized using Linear Temporal Logic (LTL). This allows us to assert and check the correctness of both refinement between goals and of realization of goals by operations. On the other hand, *agent-oriented* modeling languages, such as TROPOS [BPG+04] and $i^*$ [Yu09, Yu96], also focus on the agents that will realize these goals. A formal extension of $i^*$, featuring *commitments* and *protocols* [CDGM10b, CDGM10a, CS09, MS06], aims at checking the capabilities of agents

to ensure the satisfaction of the goals. There, goals are described using propositional logic and agents are described along with their capabilities to ensure the satisfaction of propositional formulas. Thus, the need for a treatment of the assignment problem has been identified in RE but, to the best of our knowledge, no proposition has been made until now to address it for behavioral goals, in a formalized framework.

In *Systems-of-Systems Engineering* (SoSE) [Mai98], several independent systems, made up of subsystems (agents in our parlance) interact altogether to achieve a global mission. Consider one of these systems and its set of agents $A$. Then, to investigate whether the agents in $A$ are able to ensure a given goal in the global system, one must take into account the side effects of the actions performed by the other agents (from other systems) pursuing different goals.

Finally, in Component-Based Software Engineering (CBSE) [Szy02], individual components (agents, for us) may be assembled into composite subsystems in order to fulfill requirements specifications. The capabilities of these agents are given as contracts [BJPW99]. Then knowing whether the resulting architecture indeed satisfies its specification is of major importance. Besides, identifying unsatisfied specifications can provide guidance to the engineer for adding new components. Identifying unexpected side effects (good or bad) between components is also very important.

These various examples lead us to propose the following informal characterization of the assignment problem:

**Definition 1 (Assignment Problem, Informally).** *Given a set of interacting agents, the capabilities of whom are known, given a set of goals, and given an* assignment function *mapping each goal to a coalition (*i.e. *a set) of agents, is every goal assigned to a coalition of agents who are able to ensure its satisfaction (including by benefiting from actions of other coalitions)?*

The objective of this paper is to formalize this definition and to provide a means to solve the assignment problem. In particular, notice that in this definition, what *interaction* is is left ambiguous. One of our contributions is precisely to propose multiple acceptations, each of them inducing a particular case of the problem. We call these cases *correctness criteria* for an assignment. We model the assignment problem and these criteria, and we describe a formal process to check their satisfaction for any instance.

Our approach was originally developed for agent- and goal-oriented RE [CBC11]. In this field, to the best of our knowledge, this provides the first unified formal framework addressing the satisfaction of *behavioral* goals by operation specifications and the capabilities of agents to perform these operations as required.

Checking the capabilities of agents to ensure goals also enables one to distinguish, given a set of available agents and a set of goals, those goals that the available agents *cannot* ensure. Then these goals can support the engineer in identifying *new* agents that should be introduced to fulfill all goals.

Our approach also makes it possible to characterize other sorts of interaction phenomena. Here, we stress the following:

- First, given two coalitions of agents and a goal for each coalition, we highlight *dependencies* between coalitions w.r.t. the satisfaction of their respective goals.

– Second, in an SoS for instance, we can check whether, while pursuing their own goals, agents in a system $S_1$ necessarily entail, as a *side effect* on the global system, that agents in a system $S_2$ can also ensure their own goals.

The remainder of this article is organized as follows: in Sect. 2 we introduce a minimal modeling framework, called Kore, to allow a proper representation of the situations that we wish to address. In practice, this framework can be seen as a subset of modeling languages such as Kaos or SysML. In the same section, we also introduce a minimal example which will be used in the following sections to illustrate our approach. In Sect. 3, we give a description of the assignment problem in the framework introduced by Sect. 2. To do so, we analyze various modalities of interactions between agents and we devise corresponding correctness criteria for the assignment. This way, the assignment problem is reduced to the problem of satisfaction of the evaluation criteria by an instance of Kore. Then, this problem is itself reduced to a model-checking problem for a multi-agent logic called USL in Sect. 4. Related work is discussed in Sect. 5.

## 2 A Modeling Framework for the Assignment Problem

As sketched before, a Kore model is described using a set of goals to be realized, a description of the context given by a number of *context properties*, a set of agents (considered with their *capabilities* to act on the system) and an *assignment* of the goals to (coalitions of) agents. Goals and context properties are temporal properties, so we briefly introduce in Sect. 2.1 the logic that we use to formalize them.

Let us first introduce a running example that will be used throughout this paper to illustrate our approach. In this example, we consider a resource, provided by a *provider*. Then two clients $A$ and $B$ have different needs w.r.t. this resource. As we will proceed through this article, we will envision three variations of this example.

*Example 1 ($CP_1$).* In the first version ($CP_1$), the provider can provide up to 15 units of the resource per time unit. It can also decide to which clients the resources are affected. Concretely, at each time unit it provides the clients up to 15 new units as a whole, distributed in variables $new_A$ for client $A$ and $new_B$ for client $B$. Each client is able to receive up to 15 units of the resource at a time.

### 2.1 $LTL_{\text{Kore}}$

In our setting, goals are behavioral: therefore we formalize them as well as context properties in a version of Linear Temporal Logic (LTL, [MP95]). Following [Let02,LVL02b, vL09, vL03], we describe an action (of an agent) as the modification of the values of variables describing the state of the sytem. Therefore, in our version of LTL(called $LTL_{\text{Kore}}$), atomic propositions are comparisons of integer variables and constants.

**Definition 2 (Cond).** *Let $X$ be a set of variables, the set of propositions Cond over $X$ (written Cond $(X)$) is given by the following grammar:*

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

*where $x, y \in X$, $n$ is a constant in $\mathbb{Z}$, and $\sim \in \{<, >, =, \leqslant, \geqslant\}$.*

**Definition 3** (*LTL*$_{\text{KORE}}$)**.** *Let $X$ be a set of variables, the logic $LTL_{\text{KORE}}(X)$ is the usual Linear Temporal Logic where atoms are taken from $Cond(X)$. It is generated by the following grammar:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi \quad \textit{where } p \in Cond(X).$$

Because LTL is standard knowledge in formal approaches and due to space constraints, we do not detail the formal semantics of LTL and refer the interested reader to [MP95]. Basically,an $LTL_{\text{KORE}}(X)$ formula is interpreted over discrete traces of assignments (instants) for variables in $X$. A formula $\mathbf{X}\varphi$ is true in one of these traces, at a given instant, iff $\varphi$ is true in the same trace at the next instant. A formula $\Box\varphi$ is true in a trace iff $\varphi$ is true at every instant of this trace.

*Example 2 ($CP_1$ (cont.)).* Version $CP_1$ of the *client-provider* example makes use of the following variables for every $c \in \{A, B\}$ :

- a variable $res_c$ denotes the amount of resources available for client $c$ at any time $t \geqslant 1$. This amount is the sum of:
  - the part remaining from the resource in $res_c$ at time $t-1$, denoted $old_c$
  - the resources added by *provider* in $res_c$ at transition from time $t-1$ to $t$: $new_c$
- at any time $t \geqslant 2$, $obt_c$ denotes the amount of resources obtained by $c$ from the amount in store $res_c$ at time $t-1$.

We also write $X_{\text{CP}}$ for the set of variables $\bigcup_{c\in\{A,B\}}\{res_c, old_c, new_c, obt_c\}$.

Now that $LTL_{\text{KORE}}$ has been introduced, let us delve into the concepts necessary to consider the assignment problem as we informally defined it in Sect. 1.

### 2.2 Goals

In KORE, *goals* are statement describing the behavior expected from the system. They are formalized in $LTL_{\text{KORE}}$.

*Example 3 ($CP_1$ (cont.)).* In $CP_1$, every client wishes to get a certain amount of the resource: $A$ (resp. B) wants to get at least 6 (resp. 12) units of the resource per unit of time $t \geqslant 2$. Their respective goals $g_A$ and $g_B$ are formalized as follows[3]:

$$[\![g_A]\!] \triangleq \mathbf{XX}(\Box(obt_A \geqslant 6)) \qquad [\![g_B]\!] \triangleq \mathbf{XX}(\Box(obt_B \geqslant 12))$$

### 2.3 Context Properties

*Context properties* are statements describing the system as it is (as opposed, for instance, to expected properties). In this paper:

- They may be used to specify the set of states of the system, in which case we call them *static*.

[3] We use double brackets $[\![\cdot]\!]$ to denote the formalization of an informal property.

– They may also concern the initial state of the system. In this case, they are called *initial* properties. They can be formalized using Cond only.

*Example 4 ($CP_1$ (cont.)).* In $CP_1$ we consider the following context properties:

– Static properties:
  *$Res_c$* For any client $c$, the set of resources in $res_c$ is the union of $old_c$ and $new_c$. Formally, at any time $t \geqslant 1$, the resources available are the sum of the previous ones (remaining from the value of $res_c$ at time $t-1$) and the new ones, gotten from *provider*:
  $$[\![Res_c]\!] \triangleq \Box(res_c = old_c + new_c)$$
  *$Obt_c$* At any time $t \geqslant 1$, when a client $c$ takes some resources from the set $res_c$, then at time $t+1$, the set $old_c$ is the set of resources remaining from $res_c$ at time $t$: at any time $t \geqslant 1$ the amount of resources in $old_c$ is the amount in $res_c$ minus the amount obtained by $c$ at transition from time $t-1$ to $t$:
  $$[\![Obt_{c,k}]\!] \triangleq \Box(res_c = k \rightarrow \mathbf{X}(old_c = k - obt_c))$$
  *$Posi_x$* All the variables in the example stand for the cardinality of some set of resources. Therefore they always have a non-negative value. For each $x \in X_{\mathrm{CP}}$:
  $$[\![Posi_x]\!] \triangleq \Box x \geqslant 0$$
– Initial property *$Init_x$*: in the initial state of the system, there is none of the resource anywhere. Thus, every variable is initialized to 0. For each $x \in X_{\mathrm{CP}}$:
  $$[\![Init_x]\!] \triangleq x = 0$$

### 2.4 Agents and Capabilities

*Agents* are the active entities of the system, likely to ensure or prevent the satisfaction of goals. They are described along with their *capabilities*. To define the latter, we need to consider a restriction of Cond to a fragment which characterizes finite intervals (*windows*) only:

**Definition 4 (Window conditions).** *Let $X$ be a set of variables, the language of window conditions, written $Cond^{win}(X)$, is generated by the following grammar:*

$$\varphi ::= a \leqslant x \wedge x \leqslant b \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \qquad \textit{(with } x \in X, a \leqslant b \textit{ and } a, b \in \mathbb{Z}\textit{).}$$

**Definition 5 (Capability).** *A capability Cap for an agent a is a pair of a* pre-condition *Cap.enabCond in Cond(X) and of a* window *Cap.window, in $Cond^{win}(X)$.*

The meaning of a capability is as follows: in each state where the *enabCond* holds, the corresponding agent can give to the variables appearing in *window* any values satisfying this *window*. Indeed, our modeling considers that an agent able to act upon variables is not necessarily able to give them *any* value at *any* time. The $\wedge$ connective in Def. 4, is in particular used to link bounds on different variables (see Ex. 5 below). The enabling condition defines the conditions under which an agent can use her capability and change the values of some variables, and the window bounds the set of values she can give to these variables. However, in the example we develop in this article, each window is reduced to a singleton.

*Example 5 (Capabilities for all variations of client-provider).* The capabilities for the agents in *client-provider* are given hereafter:

| Caps. $\{provide\}_{k+\ell\leqslant 15}$ for *provider*: | Caps. $\{get_c\}_{0<k\leqslant 15}$ for client $c \in \{A, B\}$: |
|---|---|
| *enabCond* : $\top$ | *enabCond* : $res_c \geqslant k$ |
| *window* : $new_A = k \wedge new_B = \ell$ | *window* : $obt_c = k$ |

### 2.5 Assignment

An assignment is simply a function from goals to coalitions (where every coalition is a set of agents). This can model different kinds of situations; for instance:

- Each agent may be pursuing her own goals (*distributed intentionality* [Yu09]). In this case the assignment models the relation between a goal and the agent(s) aiming for it.
- Or there may be a controller or an engineer, able to constrain and schedule every agent in the model, and who is responsible for the realization of the whole set of goals. In this case the assignment is an affectation, by this controller, of the available resources (the agents) to the satisfaction of the goals.

*Example 6 ($CP_1$ (cont.)).* Here we follow the second item, then *provider* is committed to the realization of both goals, so the assignment $\mathcal{A}_1$ is defined by: $\mathcal{A}_1(g_A) = \{provider, A\}$ and $\mathcal{A}_1(g_B) = \{provider, B\}$.

## 3 Evaluation Criteria for Assignment

Now that the different elements of Kore are defined, let us come back to the assignment problem and seek precise criteria modeling the notion of correctness for an assignment. We first give informal definitions for these criteria. Their formalization, which requires first the introduction of the USL framework, is given in Sect. 4.

### 3.1 Local correctness

The first version of the assignment problem we consider is the question whether each goal is assigned to a coalition able to ensure it, whatever the other agents do. We call this criterion the *local correctness* of the system under consideration. For the *client-provider* example, this is the question whether $\{provider, A\}$ is able to ensure the satisfaction of $g_A$ (whatever $B$ does) and $\{provider, B\}$ is able to ensure the satisfaction of $g_B$ (whatever $A$ does). We write $LC_{\mathcal{A}}(G)$ for the satisfaction of the local correctness of a set of goals $G$ under an assignment $\mathcal{A}$.

### 3.2 Global correctness

The criterion of local correctness is easy to understand and to check. Nevertheless it is not sufficient when, as is the case of $CP_1$, one agent is part of several coalitions being assigned different goals. Indeed, *provider* is able to take part separately in both

coalitions $\{provider, A\}$ and $\{provider, B\}$. But the local correctness does not say whether *provider* is able to take part in both coalitions *at the same time*. What *provider* has to do with the first coalition might be contradictory with what she has to do with the second coalition. To overcome this issue, we introduce a second correctness criterion, called the *global correctness*. Global correctness is satisfied if there is a *general behaviour* $b$ of all agents s.t. for each goal $g$, knowing that the coalition of agents assigned to $g$ behaves according to $b$ is enough to ensure $g$, whatever the other agents do. The notion of such a *general behaviour* is to be defined as a *multi-strategy profile* in a CGS, in Sect. 4. If the assignment $\mathcal{A}$ of a set of goals $G$ is globally correct, then we write $GC_{\mathcal{A}}(G)$.

### 3.3 Collaboration

The global correctness criterion is sufficient to ensure that each goal is assigned to a capable coalition. Nevertheless, it may require more than what is necessary. Indeed, it requires that each coalition is able to ensure its goal in a completely autonomous way (whatever the agents not in this coalition do). In certain cases, it may be necessary to soften this criterion and to admit that some given coalitions depend on others to ensure their goals. To illustrate this point, let us slightly modify our example and consider its second version, $CP_2$.

*Example 7 ($CP_2$).* It brings the following changes from $CP_1$:

- *provider* can produce up to 20 units at a time
- in the new assignment $\mathcal{A}_2$, *provider* is assigned a new goal $g_{provider}$, to produce at least 16 units of the resource per time unit. Furthermore, $g_A$ and $g_B$ are respectively assigned to $\{A\}$ and $\{B\}$.

In this second version, *provider* is able, at the same time, to ensure the satisfaction of its goal and to help $A$ and $B$. By producing, for example, at least 7 units in $new_A$ and 13 units in $new_B$, it ensures $g_{provider}$ and the global correction of the model reduced to $\{g_A, g_B\}$. In this case we say that the coalition that is assigned to $g_{provider}$ *globally collaborates* to the satisfaction of $\{g_A, g_B\}$, and we write $Coll_{\mathcal{A}_2}(g_{provider}, \{g_A, g_B\})$.

Note that a relation of *local collaboration* could also be defined a similar way.

### 3.4 Contribution

In the three criteria introduced above, we adopted the point of view of an engineer controlling every agent in the system. Thus, we considered our model as a closed system and only asked the possibility for a unique decision-maker to specify the agents so that all goals are ensured. In the case of open systems, the engineer of one system does not control the other systems, which interact with it. Then, a relevant question from the point of view of this engineer is whether the agents from the other interacting systems, by ensuring their goals, necessarily have favorable side effect on its model. This is what we call *contribution*. Let us consider a last version of our example, $CP_3$.

*Example 8 ($CP_3$).* In this version, $A$ has priority over $B$: when the provider provides resources in $new_A$ at time $t$, $A$ can get some of them, the remainder is sent to $new_B$ and then at time $t+1$, $B$ can take some. In order to encode this, we introduce a new variable *even* marking the evenness of the current time unit (it is equal to 0 at even times and equal to 1 at odd times). $A$ can act during transitions from even to odd time units (let us call them *even transitions*), and *provider* and $B$ can act during odd transitions. Again, the provider is able to produce up to 20 units of the resource per odd transition, its goal $g_{provider}$ is changed into providing at least 18 units of the resource per time unit and $g_A$ and $g_B$ are both assigned to $\{A, B\}$ in the assignment $\mathcal{A}_3$. (Fig. 1 gives an overview of the goals and the assignments in the three versions of our example.)

In this version, whatever *provider* does, if by doing so it ensures the satisfaction of $g_{provider}$ then it provides at least 18 units of the resource per time unit, enabling $A$ and $B$ to ensure the satisfaction of $g_A$ and $g_B$. We say that $g_{provider}$ *globally contributes* to $g_A$ and $g_B$ and we write $Contr_{\mathcal{A}_3}(g_{provider}, \{g_A, g_B\})$.

Again, one can also define, similarly, a relation of *local contribution*, that we do not detail here.

| | | | |
|---|---|---|---|
| $CP_1$ | $[\![g_A]\!] \triangleq \mathbf{XX}(\Box(obt_A \geqslant 6))$<br>$\mathcal{A}_1(g_A) = \{provider, A\}$ | $[\![g_B]\!] \triangleq \mathbf{XX}(\Box(obt_A \geqslant 12))$<br>$\mathcal{A}_1(g_B) = \{provider, b\}$ | |
| $CP_2$ | $[\![g_{provider}]\!] \triangleq \mathbf{X}(\Box(new_A + new_B \geqslant 16))$<br>$\mathcal{A}_2(g_{provider}) = \{provider\}$ | $[\![g_A]\!] \triangleq$ cf. $CP_1$<br>$\mathcal{A}_2(g_A) = \{A\}$ | $[\![g_B]\!] \triangleq$ cf. $CP_1$<br>$\mathcal{A}_2(g_B) = \{B\}$ |
| $CP_3$ | $[\![g_{provider}]\!] \triangleq \mathbf{X}(\Box(new_A \geqslant 18))$<br>$\mathcal{A}_3(g_{provider}) = \{provider\}$ | $[\![g_A]\!] \triangleq$ cf. $CP_1$<br>$\mathcal{A}_3(g_A) = \{A, B\}$ | $[\![g_B]\!] \triangleq$ cf. $CP_1$<br>$\mathcal{A}_3(g_B) = \{A, B\}$ |

Fig. 1: Goals and assignments in the *client-provider* example

## 4 Formal Analysis

In this section we introduce the formal framework that we use to check the correctness criteria. Basically, given a specification K conforming to the Kore framework, checking a criterion consists in knowing whether goals in K are assigned to coalitions of agents able to ensure them.

Our approach consists of reducing such a question to a model-checking problem: *does a model $\mathcal{G}$ (in the logical sense) satisfy a formula $\varphi$ ?* where: (1) the model of the possible behaviors of the system is derived from the description of agents and context properties; and (2) the formula expresses that some coalition(s) is (are) able to ensure some goal(s).

To achieve this, a logic that allows to reason about the ability of agents to ensure temporal properties is required. This is the aim of *temporal multi-agent logics*, such as ATL [AHK02], Chatterjee, Henzinger, and Piterman's Strategy Logic (SL) [CHP10] or

USL (Updatable Strategy Logic) [CBC13, CBC15], which is strictly more expressive than the former two, and which we originally proposed to address such issues.

One of the main specific features of USL is that it enables us to express situations where agents may be part of several different interacting coalitions. So, agents in USL can compose their behavior according to the different goals assigned to these coalitions. For this reason, we rely on USL in the following.

In Sect. 4.1, we briefly present USL. Then in Sect. 4.2, we present the reduction of our correctness criteria to instances of the model-checking problem for USL.

### 4.1 A temporal multi-agent logic for the formalization of Kore: USL

Let us first introduce the semantic concepts that are used in USL. Due to space constraints, we refer the reader to [CBC15] for a complete exposition of USL.

**Semantic concepts** Formulas of USL are interpreted in *concurrent game structures* (CGS), introduced in [AHK02] and then subsequently used with slight modifications in numerous works [BDCLLM09, DLLM10, MMV10, MMPV14][4]. Intuitively, a CGS is an extension of labelled transition systems dedicated to modelling multi-agent systems. In these systems, transitions are determined by the *actions* that *agents* perform. More precisely, at each state of any execution, each agent plays an action so that a transition is determined.

**Definition 6.** *A CGS is a tuple* $\mathcal{G} = \langle \mathrm{Ag}, St, s_0, At, v, Act, tr \rangle$ *where :*

- Ag *is a non-empty finite set of agents,*
- $St$ *is an non-empty enumerable set of states,*
- $s_0 \in St$ *is the initial state,*
- $At$ *is a non-empty finite set of atomic propositions*
- $v\colon St \to \mathcal{P}(At)$ *is a valuation function, to each state* $s$ *it associates the set of atomic propositions that are true in* $s$*,*
- $Act$ *is an non-empty enumerable set of actions,*
- *Let* $Dec = Act^{\mathrm{Ag}}$ *be the set of decisions,* i.e. *the set of total functions from the agents to the actions. Then* $tr : St \times Dec \to St$ *is the transition function: it decides the successor of a state, given this state and the set of actions played by the agents.*

The semantics of USL in CGSs is given by plays in a game, that are infinite sequences $(s_0, \delta_0) \cdot (s_1, \delta_1) \dots$ where for each $k \in \mathbb{N}$:

- $s_k$ is a state and $\delta_k$ is a decision
- $tr(s_k, \delta_k) = s_{k+1}$

In such a game, every agent plays w.r.t. a *multi-strategy*. A multi-strategy is a function from $St^*$ to $\mathcal{P}(Act)$: given the current history of the game, it gives a set of possible actions for any agent following it. The datum of one multi-strategy per agent in the game is called a *multi-strategy profile*.

[4] As USL builds upon SL, our definition for CGS is the one from [MMV10, MMPV14].

During the evaluation of an USL formula in a CGS, the data concerning the different multi-strategies played by the agents are stored in a *context* $\kappa$. In a context, an agent may be bound to *several* multi-strategies, which is a particularity of USL in the field of multi-agent logics. As we will see below, USL also makes use of *multi-strategy variables*. Then, a context also contains information about the instantiations of multi-strategy variables to multi-strategies. We also use the notion of *multi-strategy profile* for a coalition of agents, which consists of one multi-strategy per agent in the coalition.

Given a context $\kappa$ binding agents and variables to multi-strategies in a CGS, and given a state $s$ of this CGS, we can define the notion of *outcomes* of $s$ and $\kappa$. It is the set of executions that are possible in the CGS from $s$ if each agent plays only actions that are allowed by all the multi-strategies she is bound to in $\kappa$.

**Syntax and semantics of USL** Let us now present the syntax of USL and an intuition of its semantics. We start by defining USL pseudo-formulas. We distinguish between state pseudo-formulas (interpreted on states, whose operators deal with multi-strategies quantification and binding of multi-strategies to agents) and path pseudo-formulas (which express temporal properties).

**Definition 7 (Pseudo-formulas of USL).** *Let* Ag *be a set of agents, At a set of propositions, and X a set of multi-strategy variables. Then, the set of* USL $(Ag, At, X)$ pseudo-formulas *is generated by the following grammar (with $p \in At$, $x \in X$ and $A \subseteq \mathrm{Ag}$):*

- State pseudo-formulas: $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle x \rangle\!\rangle \varphi \mid (A \rhd x)\psi \mid (A \not\rhd x)\psi$
- Path pseudo-formulas: $\psi ::= \varphi \mid \neg\psi \mid \psi \wedge \psi \mid \mathsf{X}\psi \mid \psi \mathbf{U} \psi$

Then, well formed formulas are pseudo-formulas where every quantified multi-strategy variable is fresh w.r.t. the scope in which it is introduced. Formally:

**Definition 8 ((Well formed) formulas).** *A pseudo-formula $\varphi$ is a* (well formed) formula *iff for any sub-formula $\langle\!\langle x \rangle\!\rangle \varphi'$ of $\varphi$ and for any sub-formula $\langle\!\langle y \rangle\!\rangle \varphi''$ of $\varphi'$, $x$ and $y$ are distinct variables.*

Here we do not detail the definition for the relation of semantic satisfaction for USL ($\models_{\mathrm{USL}}$). We just give an intuition for the main operators:

- The operator $\langle\!\langle x \rangle\!\rangle$ is an existential quantifier over multi-strategies: a formula $\langle\!\langle x \rangle\!\rangle \varphi$ is true in a state $s$ of a CGS $\mathcal{G}$, under context $\kappa$, iff there is a multi-strategy $\sigma$ s.t. the formula $\varphi$ is true in $s$ and $\mathcal{G}$ under the context $\kappa$ enriched by the fact that $x$ is instantiated by $\sigma$.
- The operator $(A \rhd x)$ is a binding of agents in $A$ to the multi-strategy instantiating $x$ in the current context: a formula $(A \rhd x)\psi$ is true in a state $s$ of a CGS $\mathcal{G}$, under context $\kappa$, iff the formula $\psi$ is true in any execution in the outcomes of $s$ and $\kappa[A \oplus x]$, where $\kappa[A \oplus x]$ is the context $\kappa$ enriched with the fact that agents in $A$ are now bound to the multi-strategy instantiating $x$ in $\kappa$ (in addition to the multi-strategies they were already bound to in $\kappa$, if there are some). In USL, this set of outcomes may be empty (if there are agents bound to multi-strategies in empty intersection in the context). In such a case, the current execution stops.

– $(A \not\rhd x)$ unbinds agents in $A$ from the multi-strategy instantiating $x$ in the current context: it is interpreted in a similar way as $(A \rhd x)$, except that the binding of agents in $A$ to $x$ is deleted from the current context (instead of being added).
– The semantics of temporal operators follows the classical definition of their interpretation in possibly finite executions [EFH+03]. In the following, given a possibly finite execution $\lambda$ and an integer $i$, we write $\lambda_{\geqslant i}$ for the sequence obtained from $\lambda$ by deleting its $i$ first elements:
  - A formula $\mathsf{X}\psi$ is true in $\lambda$ iff $\lambda$ contains at least two states and $\psi$ is true in $\lambda_{\geqslant 1}$.
  - A formula $\psi_1 \mathbf{U} \psi_2$ is true in $\lambda$ iff there is $i \in \mathbb{N}$ s.t. $\lambda$ contains at least $i+1$ states, $\psi_2$ is true in $\lambda_{\geqslant i}$ and for all $0 \leqslant j < i$, $\psi_1$ is true in $\lambda_{\geqslant j}$.

### 4.2 Reduction of the assignment problem to the model-checking of USL

Using USL, we can reduce the satisfaction problem for the assignment correctness criteria from Sect. 3 to instances of model-checking. First, notice that from the description of agent capabilities in an instance $\mathsf{K}$ of Kore, and from the set of context properties for $\mathsf{K}$, one can derive a CGS $\mathcal{G}_{\mathsf{K}}$. For the sake of simplicity, we do not detail this translation here. Basically:

– Agents of $\mathcal{G}_{\mathsf{K}}$ are those of $\mathsf{K}$,
– the set of states in $\mathcal{G}_{\mathsf{K}}$ is the set of possible instantiations for the variables in $\mathsf{K}$ which respect the static context properties,
– the initial state of $\mathcal{G}_{\mathsf{K}}$ is chosen non-deterministically within those satisfying the initial context properties,
– the set of atomic propositions is the set of propositions in Cond that are used in $\mathsf{K}$ for the description of goals or for the description of agents capabilities,
– the valuation function is given by the natural evaluation of Cond formulas in variables instantiations,
– the transition function encodes the capacities of agents to change the value of variables, according to the description of their capabilities in $\mathsf{K}$.

Then, thanks to USL formulas, we can express that coalitions of agents ensure the satisfaction of the different correctness criteria for $\mathsf{K}$. The formalization of the criteria are given in Def. 9. It makes use of the following notations:

– $\overrightarrow{x}$ denotes a vector of multi-strategy variables (one can see it as a multi-strategy profile variable). Then, for any coalition of agents $A = \{a_1, \ldots, a_n\}$, the notation $(A, \overrightarrow{x})$ abbreviates the sequence $(a_1, x_{a_1}), \ldots (a_n, x_{a_n})$
– for a variable $x$, $\ulcorner x \urcorner$ is the universal quantifier over $x$. It is the dual operator of $\langle\!\langle x \rangle\!\rangle$. In other words, for any USL formula $\varphi$, $\ulcorner x \urcorner \varphi \triangleq \neg \langle\!\langle x \rangle\!\rangle \neg \varphi$.

**Definition 9 (Formalisation of the correctness criteria in USL).** *Let $\mathsf{K}$ be an instance of* Kore *with assignment $\mathcal{A}$. Let $G$ be a set of goals in $\mathsf{K}$ and let $g$ be a goal in $\mathsf{K}$ s.t.*

$g \notin G$. *Then:*

$$
\begin{aligned}
[\![LC_{\mathcal{A}}(G)]\!] &\triangleq \bigwedge_{g\in G} (\langle\!\langle \overrightarrow{x_g} \rangle\!\rangle (\mathcal{A}(g) \rhd \overrightarrow{x_g})[\![g]\!]) \\
[\![GC_{\mathcal{A}}(G)]\!] &\triangleq \langle\!\langle \overrightarrow{x} \rangle\!\rangle (\bigwedge_{g\in G} (\mathcal{A}(g) \rhd \overrightarrow{x_g})[\![g]\!]) \\
[\![Coll_{\mathcal{A}}(g,G)]\!] &\triangleq \langle\!\langle \overrightarrow{x_g} \rangle\!\rangle (\mathcal{A}(g) \rhd \overrightarrow{x_g})([\![g]\!] \wedge [\![GC_{\mathcal{A}}(G)]\!]) \\
[\![Contr_{\mathcal{A}}(g,G)]\!] &\triangleq (\langle\!\langle y_g \rangle\!\rangle (\mathcal{A}(g) \rhd \overrightarrow{x_g})[\![g]\!]) \wedge \\
&\quad \Big( \ulcorner \overrightarrow{x_g} \urcorner \big( (\mathcal{A}(g) \rhd \overrightarrow{x_g})[\![g]\!] \rightarrow ((\mathcal{A}(g) \rhd \overrightarrow{x_g})[\![GC_{\mathcal{A}}(G)]\!]) \big) \Big)
\end{aligned}
$$

Thus, for any correctness criterion $C$ from Sect. 3, and for any instance $\mathsf{K}$ of Kore, the satisfaction of $C$ by $\mathsf{K}$ is formalized by the relation $\mathcal{G}_{\mathsf{K}} \models_{\text{USL}} [\![C]\!]$.

Let us discuss on the formulas in Def. 9.

- The assignment is locally correct iff for each goal $g$, there is a multi-strategy profile s.t., by playing it, the agents to which $g$ is assigned can ensure its satisfaction. Hence, we check the satisfaction of this criterion by considering one (possibly different) multi-strategy profile per considered goal. Let us consider for instance $CP_1$ from Examples 3 and 5. We see that *provider* can play the multi-strategy $alwaysNew_A \geqslant 6$ (consisting in restricting to the choices of values for $new_A$ and $new_B$ s.t. $new_A \geqslant 6$ at any time of the execution) and, if *provider* does so, $A$ can play $alwaysObt_A \geqslant 6$ so that $g_A$ is ensured. Similarly, by playing respectively $alwaysNew_B \geqslant 12$ and $alwaysObt_A \geqslant 12$, *provider* and $B$ can ensure $g_B$. So, $(\langle\!\langle \overrightarrow{x_{g_A}} \rangle\!\rangle (A, provider \rhd \overrightarrow{x_{g_A}})[\![g_A]\!]) \wedge (\langle\!\langle \overrightarrow{x_{g_B}} \rangle\!\rangle (B, provider \rhd \overrightarrow{x_{g_B}})[\![g_B]\!])$ is true: $\mathcal{G}_{CP_1} \models_{\text{USL}} [\![LC_{\mathcal{A}_1}(\{g_A, g_B\})]\!]$.
- For the global correctness, we consider one single multi-strategy profile, which imposes on each agent to act in a coherent way. The assignment is globally correct iff there is a multi-strategy profile $\overrightarrow{x}$ s.t. for each goal $g$, if the agents in the coalition $\mathcal{A}(g)$ play according to $\overrightarrow{x}$ (in the definition, we note $\overrightarrow{x_g}$ the part of $\overrightarrow{x}$ that concerns the agents in $\mathcal{A}g$), then they ensure the satisfaction of $g$. We can easily see that $[\![GC_{\mathcal{A}_1}(G)]\!]$ is not true in $\mathcal{G}_{CP_1}$. Indeed, *provider* cannot play a multi-strategy that satisfies both $g_A$ and $g_B$ at the same time. (According to its capabilities, *provider* cannot deliver more than 15 units of the resource at a time).
- To ensure that a goal $g$ globally collaborates to a set of goals $G$, we need a multi-strategy profile $\overrightarrow{x}$ s.t., if followed by the agents in $\mathcal{A}(g)$, $\overrightarrow{x}$ ensures at the same time that:
  - $g$ is satisfied
  - the evolution of the model is constrained in such a way that the assignment $\mathcal{A}$ becomes globally correct for the set of goals $G$.

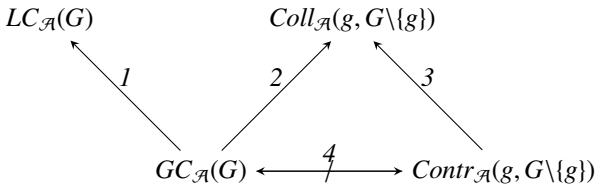
  According to this definition, the example $g_{prov}$ globally collaborates to $g_A$ and $g_B$ in $CP_2$ (see Sect. 3.3). Indeed, since it may produce up to 20 units of the resource per time unit, *provider* can play a multi-strategy that will allow both $A$ and $B$ to ensure their respective goal. Furthermore, recall that $\mathcal{A}_2(g_A)$ is reduced to $\{A\}$ and $\mathcal{A}_2(g_B)$ is reduced to $\{B\}$ and observe that $CP_2$ is not globally correct: to be able to ensure their goals, $A$ and $B$ depend on the multi-strategy played by *provider*.
- The contribution relation is an universally quantified variant of the collaboration : a goal $g$ globally contributes to a set of goals $G$ iff

- the agents in $\mathcal{A}(g)$ are able to ensure $g$,
- for any multi-strategy profile $\overrightarrow{x_g}$ that makes $\mathcal{A}(g)$ ensure $g$, $\overrightarrow{x_g}$ also makes $\mathcal{A}(g)$ constrain the evolution of the system in a way that $G$ becomes globally correct.

In $CP_2$, by playing, for example, the multi-strategy consisting in setting $new_A$ to 5 and $new_B$ to 11, *provider* ensures its goal of producing at least 16 units per time unit, but it prevents $A$ and $B$ to ensure both goals $g_A$ and $g_B$. On the other hand, in $CP_3$, the satisfaction of $g_{provider}$ (providing at least 18 units of the resource) by the provider allows $A$ an $B$ to ensure their goals, provided the new assignment of both $g_A$ and $g_B$ to $\{A, B\}$. So $[\![Contr_{\mathcal{A}_3}(g_{prov}, \{g_A, g_B\})]\!]$ is true in $\mathcal{G}_{CP_3}$.

**Theorem 1.** *The possible entailment relations between our correctness criteria for the assignment are given in the following figure, where arrows* 1, 2 *and* 3 *represent a strict entailment relation, and the crossed out arrow* 4 *means that there is no entailment between* $GC_{\mathcal{A}}(G)$ *and* $Contr_{\mathcal{A}}(g, G\backslash\{g\})$*, in either direction. Each arrow should be read by universally quantifyfing the assignment* $\mathcal{A}$*, the set of goals* $G$ *and any goal* $g \in G$*. For example:*

- ***Entailment*** *For any instance* $\mathsf{K}$ *of* Kore *with assignment* $\mathcal{A}$ *and for any subset* $G$ *of goals in* $\mathsf{K}$*, for any* $g \in G$*, if* $\mathcal{G}_{\mathsf{K}} \models_{USL} [\![GC_{\mathcal{A}}(G)]\!]$ *then* $\mathcal{G}_{\mathsf{K}} \models_{USL} [\![Coll_{\mathcal{A}}(g, G\backslash\{g\})]\!]$.
- ***Strictness*** *It is not true that for any instance* $\mathsf{K}$ *of* Kore *with assignment* $\mathcal{A}$ *and for any subset* $G$ *of goals in* $\mathsf{K}$*, for any* $g \in G$*, if* $\mathcal{G}_{\mathsf{K}} \models_{USL} [\![Coll_{\mathcal{A}}(g, G\backslash\{g\})]\!]$ *then* $\mathcal{G}_{\mathsf{K}} \models_{USL} [\![GC_{\mathcal{A}}(G)]\!]$.




*Proof (sketch).* Each item in this proof sketch refers to the arrow in figure above that has the corresponding label.

1. Entailment : straightforward. Strictness: as seen in Sect. 4.2, $CP_1$ provides a counterexample.
2. Entailment: suppose there is a general multi-strategy profile $\overrightarrow{x}$ s.t., by playing it, every coalition ensures its goal. Then, by playing along $\overrightarrow{x}$, the agents in $g$ ensure at the same time the satisfaction of $g$ and the global correction of the model reduced to $G \setminus \{g\}$. Strictness: as seen in Sect. 4.2, $CP_2$ is a counterexample for the converse.
3. Entailment : straightforward. Strictness: goal $g_{provider}$ in $CP_2$ provides a counterexample: by playing multi-strategies $allwaysNew_A = 5$ and $allwaysNew_B = 11$, *provider* ensures $g_{provider}$ but prevents the satisfaction of $g_A$ and $g_B$ by the other agents.
4. Left to right: $CP_3$ satisfies $Contr_{\mathcal{A}_3}(g_{provider}, GC(\{g_A, g_B\}))$ but, to be able to ensure their goals, $A$ and $B$ depend on the satisfaction of $g_{provider}$ by *provider*, so $GC_{\mathcal{A}_3}(\{g_A, g_B\})$ is not true. Right to left: consider a minimal example where *provider* is able to provide 5 units of the resource per time unit, and is assigned both goals $g_-$ to provide 2 units, and $g_+$ to provide 4 units. This model is globally correct but $g_-$ does not contribute to $\{g_+\}$. □

## 5 Related Work

This work was initially developed in the context of Requirements Engineering [CBC11] and took inspiration from the state-of-the-art in this domain, in particular from Kaos [vL09, LVL02b, Let02]. In this method, goals are gradually refined until reaching so-called requirements. Then, agents are assigned the responsibility of realizing the latter by relying on operations (our agents are directly assigned goals to simplify the presentation). In some developments of Kaos, a notion of controllable and monitorable conditions [LVL02a, vL04] is used as a criterion of satisfiability of realization: an agent can perform an operation if it monitors the variables in its pre-conditions and controls the ones in its post-conditions. So, in Kaos and contrary to Kore, (1) capabilities are not conditioned by the state of the system; and (2) agents interactions are not analyzed.

Another important RE approach is Tropos [CDGM10a, CDGM10b, MS06, CS09]. In this line of work, a notion of *role* is introduced which gathers a set of specifications to be satisfied by the system. The two notions of agents and roles are then confronted. The adequacy between them is examined using propositional logic: roles are described through commitments and agents may ensure these depending on their capabilities. Considerations on time and interactions between agents are only led using natural language. Thus the method makes the verification of questions of the sort: "can agent $a$ ensure transition $tr$?" possible. But the possible interactions between agents, as modifications of the common environment, are not considered formally. Kore precisely aims at unifying the multi-agent and behavioral aspects.

On the logical side, ATL and ATL* [AHK02] consider the absolute ability of coalitions to ensure propositions whatever other agents do. But there is no contextualization w.r.t. the strategies followed by different agents. More recently, this contextualization was considered for ATL* [BDCLLM09]. This proposition only uses implicit quantification over strategies for coalitions, preventing from considering different strategy quantifications for a given coalition. This problem was also tackled in SL [MMV10] where quantification over strategies is made using explicit variables. Our logic USL was first developed in [CBC13, CBC15]. Its syntax is inspired by that of SL, but it contains in addition treatment for the composition of several multi-strategies for a given agent.

## 6 Conclusion and Future Work

In this article, we proposed a framework to model the assignment of behavioral goals to agents, described with capabilities. Then, we addressed the evaluation of such an assignment, informally referred to as the *assignment problem*. We proceed in two steps:

- Rather than defining one criterion that would provide a unique "yes or no" answer, we think it is more relevant to define several correctness criteria, each involving a different level of interaction between agents. We compared these criteria through a logically defined entailment relation between them.
- We provide a formalization of the different criteria using a temporal multi-agent logic, USL, and we reduce the verification of these criteria to the model-checking problem of this logic.

As a future work, the relevance of new correctness criteria for the assignment could be investigated. A direction would be to develop a formal language dedicated to the specification of criteria, using the satisfaction of goals by the coalitions they are assigned to as atoms, and the relations between these goals as operators. Thanks to such a language, we could extend and refine the criteria that can be checked in KORE.

Another direction is to study dependence relations between the multi-strategies that are played by the different coalitions. In the contribution relation for example, a coalition $A_1$ is able to find a favorable multi-strategy profile, whatever another coalition $A_2$ does (because of the nesting of multi-strategy quantifiers). In other words, $A_1$ knows the whole multi-strategy profile chosen by $A_2$ when choosing its own multi-strategy profile, which is a very strong assumption. However, it is possible in USL to characterize several forms of independence of $A_1$'s multi-strategy profile with respect to $A_2$'s multi-strategy profile, so that this question could be integrated in the definition of new correctness criteria.

In [CBC15], we proved that the model-checking problem for USL is decidable, but does not support any elementary bound. Nevertheless, we have a strong conjecture stating that the restriction of USL to memoryless multi-strategies is decidable in PSPACE. Thus, restricting to memoryless multi-strategies appears as an important condition for a tractable use of our proposition. Then, research should be led in order to further characterize the class of systems for which a memoryless semantics is adequate.

Finally, it can happen that some goals are not fully achievable by the agents they are assigned to. Especially, so called *soft goals* don't have any clear cut satisfaction criterion. Then, considering and searching the best multi-strategies with regards to these goals would rise further analyses. Different notions of optima are indeed expressible in USL and could be used for defining different notions of optimal multi-strategies.

## References


AHK02. Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

BDCLLM09. T. Brihaye, A. Da Costa Lopes, F. Laroussinie, and N. Markey. ATL with strategy contexts and bounded memory. *Logical Foundations of Computer Science*, pages 92–106, 2009.

BJPW99. Antoine Beugnard, Jean-Marc Jézéquel, Noël Plouzeau, and Damien Watkins. Making components contract aware. *Computer*, 32(7):38–45, 1999.

BPG+04. 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.

CBC11. Christophe Chareton, Julien Brunel, and David Chemouil. A formal treatment of agents, goals and operations using alternating-time temporal logic. In *Brazilian Symposium on Formal Methods (SBMF)*, pages 188–203, 2011.

CBC13. Christophe Chareton, Julien Brunel, and David Chemouil. Towards an Updatable Strategy Logic. In *Proc. 1st Intl WS on Strategic Reasoning SR*, 2013.

CBC15. Christophe Chareton, Julien Brunel, and David Chemouil. A logic with revocable and refinable strategies. *Inf. Comput.*, 242:157–182, 2015.

CDGM10a. 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.

CDGM10b. A.K. Chopra, F. Dalpiaz, P. Giorgini, and J. Mylopoulos. Reasoning about agents and protocols via goals and commitments. In *Proc. of the 9th Intl Conf. on Autonomous Agents and Multiagent Systems-Volume 1*, pages 457–464, 2010.

CHP10. Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Inf. & Comp.*, 208(6):677–693, 2010.

CS09. A.K. Chopra and M.P. Singh. Multiagent commitment alignment. In *Proc. of The 8th Intl Conf. on Autonomous Agents and Multiagent Systems-Volume 2*, pages 937–944, 2009.

DLLM10. Arnaud Da Costa Lopes, François Laroussinie, and Nicolas Markey. ATL with strategy contexts: Expressiveness and model checking. In *IARCS Annual Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8, pages 120–132, 2010.

EFH+03. Cindy Eisner, Dana Fisman, John Havlicek, Yoad Lustig, Anthony McIsaac, and David Van Campenhout. Reasoning with temporal logic on truncated paths. In *Computer Aided Verification*, pages 27–39, 2003.

Let02. E. Letier. *Reasoning about Agents in Goal-Oriented Requirements Engineering*. PhD thesis, Université Catholique de Louvain, 2002.

LVL02a. E. Letier and A. Van Lamsweerde. Agent-based tactics for goal-oriented requirements elaboration. In *Proc. of the 24th Intl Conf. on Software Engineering*, pages 83–93. ACM, 2002.

LVL02b. E. Letier and A. Van Lamsweerde. Deriving operational software specifications from system goals. In *Proc. of the 10th ACM SIGSOFT symposium on Foundations of software engineering*, page 128. ACM, 2002.

Mai98. Mark W. Maier. Architecting principles for systems-of-systems. *Systems Engineering*, 1(4):267–284, 1998.

MMPV14. Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y Vardi. Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic (TOCL)*, 15(4):34, 2014.

MMV10. Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Reasoning about strategies. In *IARCS Annual Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8, pages 133–144, 2010.

MP95. Zohar Manna and Amir Pnueli. *Temporal verification of reactive systems - safety*. Springer, 1995.

MS06. A. Mallya and M. Singh. Incorporating commitment protocols into Tropos. *Agent-Oriented Software Engineering VI*, pages 69–80, 2006.

Szy02. Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, 2nd edition, 2002.

vL03. Axel van Lamsweerde. From system goals to software architecture. In *Formal Methods for Software Architectures*, pages 25–43, 2003.

vL04. Axel van Lamsweerde. Elaborating security requirements by construction of intentional anti-models. In *ICSE*, pages 148–157, 2004.

vL09. Axel van Lamsweerde. *Requirements Engineering - From System Goals to UML Models to Software Specifications*. Wiley, 2009.

Yu96. Eric Siu-Kwong Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, University of Toronto, Toronto, Ont., Canada, Canada, 1996. UMI Order No. GAXNN-02887 (Canadian dissertation).

Yu09. Eric S. K. Yu. Social modeling and *i**. In *Conceptual Modeling: Foundations and Applications*, pages 99–121, 2009.