Abstract

In this paper we are interested in two points in Requirement Engineering: modelization of requirements and distribution of requirements. We represent requirements as orders on possible situations, which allows one to have a representation of less ideal cases. Using this formalization, we distribute requirements among a group of executing agents, for which an agency model is defined.

1. Introduction

Requirements Engineering (RE) describes the process which leads to the production of specifications, or requirements, about a software to be build. In [10], Pohl presents RE as a three-dimensions space: specification, representation, agreement. The specification dimension represents the degree of specification which varies from opaque (the initial needs) to complete. The representation dimension deals with the different ways to represent knowledge about the software and its environment. There are three categories of representations: informal (natural language, graphics...), semi-formal (SADT, UML...) and formal (specification languages, knowledge languages...). The agreement dimension represents the fact that all the participants in the RE process must agree on the specifications.

All those dimensions do not deal with the moment when we will begin the implementation of the software. According to us, adding a fourth dimension is interesting. This dimension concerns the distribution of requirements among a group of executing agents, those who will effectively build the software. In [4], Easterbrook notices that validation is an important part of requirements engineering, but cannot be formalized. Formalizing the distribution of requirements could be a way to “pre-validate” those requirements at a design level. If the requirements cannot be distributed entirely, the software cannot be build. Furthermore, this could serve as a basis to build a plan for development.

In this paper, we are only interested in the representation dimension and the distribution of requirements among a group of executing agents. In this work, we occult the agreement phase and we consider the requirements as elicited, i.e. the persons involved in the RE process agree on the requirements representation. Notice that eliciting requirements can be viewed as a merging of the participants views. There are different kinds of merging, and a judicious one for requirements would be majority merging [6], in which the merging process elicits a requirements if it is supported by the majority of the participants.

More, in this present paper we are not interested in the evolution of requirements. If it is proved that an agreement cannot be achieved, a negotiation phase must be initiated, after which the participants views (including the executing agents views) should be updated [13, 9].

The first point developed in this paper is the representation of requirements. Consider a requirement emitted by a participant of the form “I want the software to verify property $A$”. We think that when such a sentence is emitted by a participant, it may not only think about the property (what is ideal for it), but he may think about the less ideal cases. For instance, it may think that, if the situation where $A$ is not verified occurs, then it prefers that $B$ is verified. An adequate formalism to represent requirements must offer a representation of both ideal and less ideal situations. Our formalization is based on the logic $CO^*$, a logic of preferences developed by Craig Boutilier [1], because its semantics allows to represent orders among situations.

The second point we study in this paper is the distribution of requirements among a group of executing agents. The distribution process consists in calculating the goals of the executing agents from the set of requirements. To distribute requirements among agents, we need a model of the agent. The main approach to capture the notion of rationality of an agent is the BDI (for Belief, Desire, Intention)
architecture [11, 12]. In this work, we consider only three notions to modelize the agents: their knowledge of the real world, which is supposed to be common to all agents, their abilities and their commitments. To calculate the goals of the group of executing agents and of each agent, we use the logic $CO^*$ again.

This paper is organized as follows. In the second section, we present the logic $CO^*$ and our representation of requirements. In section 3, we study the mechanism of goal derivation for one agent. Then in section 4, we extend this mechanism to a group of agents by redefining first the notion of goal for a group of agents. We define next the notion of commitment for an agent, how to deduce an agent’s effective goals from a set of requirements and we study an example. Finally, we present our conclusions and future work prospects in section 5.

2. Expressing Requirements with $CO^*$

In this section, we present $CO^*$, a logic of preferences developed by Craig Boutilier. We only recall its semantics, the axiomatic being described in [1]. Then we present our formalization or requirements using $CO^*$.

2.1. The logic $CO^*$

Boutilier assumes a propositional bimodal language over a set of atomic propositional variables $PROP$, with the usual connectives and two modal operators denoted $\Box$ and $\Diamond$. The semantic of $CO^*$ is based on models of the form $M = \langle W, \leq, \phi \rangle$ where $W$ is a set of worlds, $\phi$ is a valuation function which associates any propositional letter with a formula. $\leq$ is a total preorder on worlds (i.e. a reflexive, transitive and connected binary relation over $W$). $v \leq w$ means that $v$ is at least as preferred as $w$. A constraint is imposed on the $CO^*$ models. We recall it after giving the definition of the valuation of a formula.

Let $M = \langle W, \leq, \phi \rangle$ be a $CO^*$ model. The valuation of a formula in $M$ is given by the following definition:

**Definition 1.**
- $M \models \alpha$ iff $\forall w \in W M \models_w \alpha$ for any formula $\alpha$
- $M \models_w \alpha$ iff $w \in \phi(\alpha)$ for any propositional letter $\alpha$
- $M \models_w \neg \alpha$ iff $M \not\models_w \alpha$ for any formula $\alpha$
- $M \models_w (\alpha_1 \land \alpha_2)$ iff $M \models_w \alpha_1$ and $M \models_w \alpha_2$, if $\alpha_1$ and $\alpha_2$ are formulas
- $M \models_w \Box \alpha$ iff for all $v$ so that $v \leq w$, $M \models_v \alpha$
- $M \models_w \Diamond \alpha$ iff for all $v$ so that $w < v$, $M \models_v \alpha$

Thus, $\Box \alpha$ is true in a world $w$ iff $\alpha$ is true in all the worlds at least as preferred as $w$. And $\Diamond \alpha$ is true in all the worlds less preferred than $w$. As usual, the dual operators $\Diamond$ and $\Box$ are defined by $\Diamond \alpha \equiv_{df} \neg \Box \neg \alpha$ and $\Box \alpha \equiv_{df} \neg \Diamond \neg \alpha$.

Boutilier also defines $\neg \alpha \equiv_{df} \Box \alpha \land \Diamond \alpha$ and $\Diamond \alpha \equiv_{df} \alpha \lor \Diamond \Diamond \alpha$.

The constraint imposed on every $CO^*$ model $M$ is the following: for every satisfiable formula $\varphi$ of $PROP$, there is at least one world $w$ so that $M \models_w \varphi$.

For instance, let us consider a model $M$ consisting of 4 worlds $w_1, w_2, w_3$ and $w_4$ ordered as follows:

$$\begin{array}{c}
\frac{w_1}{a/b} \\
\frac{w_2}{a/b} \\
\frac{w_3}{a/b} \\
\frac{w_4}{a/b}
\end{array}$$

Then $M \models_{w_2} \neg b$ since every world at least as preferred as $w_2$ satisfies $b$.

**Definition 2.** As usual, we say that $M$ satisfies a formula $\alpha$ iff $M \models \alpha$. Let $E$ be a set of formulas and $\alpha$ a formula of $CO^*$. We say that $\alpha$ is derived (or deduced) from $E$ iff any model $M$ which satisfies $E$ also satisfies $\alpha$. We note it $E \models \alpha$.

2.2. Expressing preferences

In order to express conditional or absolute preferences, Boutilier defines a conditional connective $\langle \text{true} \rangle$ by the following definition:

**Definition 3.** $I(b|a) \equiv_{df} \neg a \lor \Box (a \land \Box (a \rightarrow b))$ et $I(a) \equiv_{df} I(\text{true}|a)$ where $\langle \text{true} \rangle$ is any propositional tautology.

We can interpret $I(b|a)$ by “if $a$ is true then we prefer $b$ to be true”. $I(a)$ means that the most preferred worlds are $a$-worlds.

We can wonder why we do not use the operator $I(\text{true}|\Box a)$ to express requirements. Our point of view is that $I(\text{true}|\Box a)$ is not strict enough to deal with requirements. For instance:

$$\begin{array}{c}
\frac{w_1}{a/b} \\
\frac{w_2}{a/b} \\
\frac{w_3}{a/b} \\
\frac{w_4}{a/b}
\end{array}$$

is a model of $I(a)$ although the least preferred world is $a$.

Boutilier also defines a notion of strict preference, i.e. if some proposition $a$ is more desirable than its negation, we can assert $\Box (a \rightarrow \Box a)$. This modelization corresponds to the notion of requirements. Unfortunately, two
strict preferences cannot be mixed, i.e. there is no model for 
\( \lnot (a \rightarrow \square a) \land \lnot (b \rightarrow \square b) \) for instance.

\( CO^* \) is a tool to express orders on worlds. Preferences, as defined by Boutilier, are not sufficient to modelize requirements. We present in the next sections our modelization of requirements. We focus on two type of requirements: absolute requirements and conditional requirements.

2.3. Absolute Requirements

First, we want to modelize absolute requirements, i.e. sentences of the form “I want the software to satisfy the property \( q \)”. In our approach, we use ordered worlds, so we need first to explain intuitively how we can represent absolute requirements with an order on worlds.

According to us, the signification of the sentence “I want the software to satisfy the property \( q \)”, is the following: for every \( \lnot a \) world (i.e. a world that satisfies \( \lnot a \) \( w_{\lnot a} \):

- all the less preferred worlds are \( \lnot a \) worlds
- there is a \( a \) world \( w_a \) at least as preferred as \( w_{\lnot a} \) so that all the worlds at least as preferred as \( w_a \) are \( a \) worlds.

We note an absolute requirement on the property \( a \) \( E(a) \). The formal definition is:

Definition 4.

\[
E(a) \equiv_{df} \Box (\lnot a \rightarrow (\Box \lnot a \land \square a))
\]

2.4. Conditional Requirements

In this work, we call “conditional requirement” a sentence of the form “I want the software to satisfy property \( a \) if \( b \) is true”, noted \( E(a|b) \). The easiest modelization would be \( E(a|b) \equiv_{df} E(b \rightarrow a) \), but this definition prevent us from expressing for instance \( E(a|b) \land E(\lnot a|\lnot b) \). If we want a model \( M \) to reflect the sentence \( E(a|b) \), we want the worlds to be ordered as following:

- there is a world that verifies \( (a \land b) \)
- all the worlds less preferred than this world are \( (\lnot a \lor \lnot b) \) worlds
- all the worlds at least as preferred as this world verify \( (b \rightarrow a) \)

This representation is intuitively correct (a world that verifies \( (a \land b) \) cannot be less preferred than \( a \) \( (\lnot a \land b) \) world... and we can combine multiple conditional requirements. The formal definition follows.

Definition 5.

\[
E(a|b) \equiv_{df} \Box ((a \land b) \land \Box (\lnot b \lor \lnot a) \land \square (b \rightarrow a))
\]

For instance, let us consider the model \( M \):

\[
\begin{align*}
\text{world } w_1 & \approx \text{world } w_2 \\
\text{world } w_3 & \approx \text{world } w_4 < \text{world } w_3 < \text{world } w_1 < \text{world } w_4
\end{align*}
\]

\( M \) is a model of \( E(a|b) \land E(\lnot a|\lnot b) \). This model satisfies \( E(a|b) \) because:

- \( M \models_{w_1} a \land b \\
- \( M \models_{w_2} b \rightarrow a \\
- \( M \models_{w_3} \lnot a \) and \( M \models_{w_4} \lnot b \\
- w_1 < w_3 \) and \( w_1 < w_4 \)

2.5. Properties of \( E(\neg) \) and \( E(\neg\neg) \)

In this section, we examine some properties of the two operators \( E(\neg) \) et \( E(\neg\neg) \).

Property 1.

\[
\models E(a) \rightarrow \neg E(\lnot a)
\]

\[
\models E(a) \land E(b) \rightarrow E(a \land b)
\]

The operator \( E(\neg) \) can be considered as a \( (KD) \) operator [3].

Property 2.

\[
\models E(a|b) \land E(b) \rightarrow E(a)
\]

This property can be viewed as “deontic detachment” [2].

Property 3.

\[
\models E(a) \rightarrow I(a)
\]

\[
\models E(a|b) \rightarrow I(a|b)
\]

The proof is immediate. The property is intuitively correct, because a preference is weaker than a requirement.

Property 4. Requirements cannot be inconsistent. In other words \( \models \neg (E(a) \land E(\lnot a)) \) and \( \models \neg (E(a|b) \land E(\lnot a|c)) \) if \( \models c \rightarrow b \).

Inconsistency management is a very important field of RE, because tolerating temporary inconsistent requirements offers more alternatives to obtain the final specifications [7, 8, 5]. As we consider that we want to represent a set of already elicited requirements, this property is not awkward.
2.6. Example

Let us consider the following example. We want to develop a database with a server. We want also a corresponding HTML page to access this database. But, if there is no database, we do not create the web page, because this is no worth creating a web page to access it. The absolute requirements here are: there is a database. The conditional requirement we can emit are if there is no database, there is no web page and if there is a database, there is a web page.

We represent “there is a database” by the propositional letter $db$ and “there is a web page to access it” by $wp$. We model those requirements by the set $\Sigma = \{E(db), E(\neg wp \land \neg db), E(wp \land db)\}$. Notice that if we cannot create the web page, there is no explicit requirement about the database. The model of $\Sigma$ is:

$$
\begin{align*}
& db & < & \neg wp & < & \neg db & < & \neg wp
\end{align*}
$$

Notice that in this case, there is only one model of $\Sigma$, but there could be more models in other cases.

3. Deducing goals from a set of ordered worlds

The second point we study in this paper is the distribution of goals among a group of executing agent from a set of requirements. As we saw before, we can express requirements as orders on worlds, i.e $CO^*$ models. A set of requirements can be represented as a set of ordered worlds (see example 2.6). We want now derive goals from this set of ordered worlds. In this section, we present the technics of Boutilier in [2] to derive goals.

3.1. Ideal goals

To determine its goals, an agent must have some knowledge about the real world. Boutilier introduces $KB$, a finite and consistent set of propositional letters, to represent the knowledge of the agent about the real world. We call $KB$ a knowledge base. Given $KB$ and a $CO^*$ model, the ideal situations are characterized by the most preferred worlds which satisfy $KB$. The formal definition is (notice that, unlike Boutilier, we focus on semantics):

**Definition 6.** Let $\Sigma$ be a set of requirements and $M_1 \ldots M_n$ all its possible models\(^1\). Let $KB$ be a knowledge base and $Cl(KB) = \{\alpha : KB \models \alpha\}$.

For $j \in \{1 \ldots n\}$, $M_j$ is the restriction of $M_j$ to $Cl(KB)$ worlds\(^2\).

\(^1\)Remember that each model is a set of ordered worlds
\(^2\)Obtained by considering $M_j$ and deleting any worlds which does not satisfy $Cl(KB)$

An ideal goal derived from $\Sigma$ is a propositional formula $\alpha$ so that: $\forall j \in \{1 \ldots n\} M_j \vdash I(\alpha)$

Let us resume our previous example, i.e. $\Sigma = \{E(wp), E(\neg wp \land \neg db), E(wp \land db)\}$. Suppose that there is a database, i.e. $KB = \{\neg db\}$. In this case, the only restricted model of $\Sigma$ is:

$$
\begin{align*}
& \neg db & < & \neg wp
\end{align*}
$$

So in the ideal situations, $\neg wp$ is true. Ideally, the agent has not to create the web page.

3.2. Controlability of propositions

By the previous definition, every formula $\alpha$ which satisfies $\forall j \in \{1 \ldots n\} M_j \models I(\alpha)$ is a goal for the agent. But, as Boutilier notes, this definition is fair only if $KB$ is fixed. If the agent can change the truth of some elements in $KB$, ideal goals as previously defined may be too restrictive. If there is no database (i.e. $KB = \{\neg db\}$) but the agent can build the database (this a quite reasonable assumption), then we cannot derive the agent goals from $\neg db$. So, for deriving goals, Boutilier suggests considering only the formulas whose truth cannot be changed by the agent’s actions.

Furthermore, as Boutilier notes, even if the agent can emit preferences about propositions over which he/she has no control (he/she may prefer that it is sunny for instance), calling those propositions a goal is unreasonable.

To capture such distinctions, Boutilier introduces a simple agent’s ability model to demonstrate its influence on goals. He partitions the atomic propositions in two classes: $P = C \cup \overline{C}$, in which $C$ is the set of atomic propositions that the agent can control (i.e. the agent can change the truth value of those propositions) and $\overline{C}$ is the set of atomic propositions that the agent cannot control.

Then he generalizes this notion as follows:

**Definition 7.** For any set of atomic variables $P$, let $V(P)$ be the set of truth assignments to this set. If $v \in V(P)$ and $w \in V(Q)$ for disjoint sets $P$ and $Q$, then $v; w \in V(P \cup Q)$ denotes the obvious extendede assignment. The neutral element of $\overline{C}$ is $V(\phi)$ which is the empty truth assignment.

**Definition 8.** A proposition $\alpha$ is controlable iff, for all $u \in V(\overline{C})$, there is some $v \in V(C)$ and some $w \in V(C)$ so that $v; u \models \alpha$ and $w; u \models \neg \alpha$.

A proposition $\alpha$ is influenceable iff, for some $u \in V(\overline{C})$, there is some $v \in V(C)$ and some $w \in V(C)$ so that $v; u \models \alpha$ and $w; u \models \neg \alpha$.

A proposition $\alpha$ is uninfluenceable iff it is not influenceable.
For instance, of $a \in C$ and $b \in \overline{C}$, then $a \land b$ is influenceable but not controllable.

**Definition 9.** The influenceable belief set of an agent is defined by $UI(KB) = \{ \alpha \in Cl(KB) : \alpha$ is influenceable$\}$.

In a first part of his work, Boutilier considers that $UI(KB)$ is complete, i.e. the truth value of all uncontrollable atoms is known. We will limit our work to this case. Under this hypothesis, Boutilier defines the notion of CK-goal as follows:

**Definition 10.** Let $\Sigma$ be a set of requirements and $M_1 \ldots M_n$ all its possible models. Let $KB$ be a knowledge base such that $UI(KB)$ is complete.

For $j \in \{1 \ldots n\}$, $M_j^{CK}$ is the restriction of $M_j$ to $UI(KB)$ worlds.

A CK goal derived from $\Sigma$ is a proposition $\alpha$ so that:

$\forall j \in \{1 \ldots n\} M_j^{CK} \models I(\alpha)$ and $\alpha$ is controllable.

Let us resume our example. Suppose now that there is no database and that nobody can build the database (nobody has the ability to build the database). So $\neg db \in KB$ and $db \in \overline{C}$. The restricted models $M_j^{CK}$ of $\Sigma$ are:

\[
\begin{array}{c}
\neg db \\
\neg wp
\end{array}
\]

Suppose that the agent can create the web page, i.e. $wp \in C$. In this case, $\neg wp$ is a CK goal derived from $\Sigma$: the agent will have not to create the web page.

### 4. Distribution of a set of requirements among a set of executing agents

We will now consider a group of executing agents $A = \{a_1, \ldots, a_m\}$. As defined previously, if we want to determine the CK goals of $A$, we must know which propositions are controllable by $A$. Most of the time, we only know the controllable atoms for each agent $a_k$. In order to determine the formulas controllable by $A$, we must first extend the controllability notion to a group of agents.

#### 4.1. Extension of controllability

We will extend the notion of controllability introduced by Boutilier (a dichotomy over the set of propositional letters) by partitioning the set of atoms in two classes for each agent:

**Definition 11.** For each agent $a_k \in A$, we partition the atoms in two classes: $C_{a_k}$ that represents the atoms which are in the controllability domain of $a_k$ and $\overline{C}_{a_k}$ that represents the atoms which are not in the controllability domain of $a_k$.

We keep the same extension to formulas as Boutilier. Notice that the agents’ controllability domain are not necessary disjoint. Two different agents may be competent to achieve the same action.

We express now the relation between agent’s controllability and group’s controllability.

**Definition 12.** A proposition $\alpha$ is semi-controllable by a group $A$ iff $\exists A' \subseteq A$ so that $\forall a_k' \in A' \exists \neg \phi_s$ controllable proposition by $a_k'$ so that $\vdash (\bigwedge_{a_k' \in A'} \phi_s) \rightarrow \phi$.

A proposition $\varphi$ is controllable by a group $A$ iff $\varphi$ is semi-controllable by $A$ and $\neg \varphi$ is semi-controllable by $A$.

Let us take an example. Let $\{a_1, a_2\}$ be a group of agents so that $p$ is controllable by $a_1$ and $r$ is controllable by $a_2$. What is the controllability of $(p \lor q) \land (r \lor s)$ for $\{a_1, a_2\}$?

On one hand, as $(p \land r) \rightarrow (p \land q) \land (r \lor s)$, and as $p$ is controllable by $a_1$ and $r$ is controllable by $a_2$, $(p \lor q) \land (r \lor s)$ is semi-controllable by $\{a_1, a_2\}$. On the other hand, $(\neg p \land \neg q) \lor (\neg r \land \neg s)$ is not semi-controllable by $\{a_1, a_2\}$ (you can choose $\phi(q) = \top$ and $\phi(s) = \top$), so $(p \lor q) \land (r \lor s)$ is not controllable by $\{a_1, a_2\}$.

We can now introduce the definition of a CK goal for a group $A$. We first clarify the definition of $KB$ and $UI(KB)$.

**Definition 13.** $KB$ is a finite and consistent set of propositional formulas. $UI(KB) = \{ \alpha : KB \models \alpha \land \alpha$ is not controllable by $A \}$. $KB$ is supposed to be complete.

**Definition 14.** Let $\Sigma$ be a set of requirements, $M_1 \ldots M_n$ all its possible models and $\forall j \in \{1 \ldots n\}$, $M_j^{CK}$ be defined as in definition 10.

A CK goal derived from $\Sigma$ for $A$ is a propositional formula $\alpha$ so that:

- $\alpha$ is controllable by $A$
- $\forall j \in \{1 \ldots n\} M_j^{CK} \models I(\alpha)$

If we want to determine the effective tasks that each executing agent must achieve in order to realize the group’s goals, we have to modelize the commitments of the agents. Let us resume our example: if we have two agents, $a_1$ who can create the database and $a_2$ who can create the web page, $a_2$’s task about the web page depends on $a_1$’s commitment about the database (if $a_1$ undertakes to not create the database for instance).
4.2. Agents commitments

Let \( a_k \) be an agent of \( A \). To represent \( a_k \)'s commitments, we use two sets of controllable literals: \( \text{Eng}_+(a_i) \) and \( \text{Eng}_-(a_i) \). They are defined as follows:

- if \( l \) is a literal and controllable by \( a_k \), \( l \in \text{Eng}_+(a_k) \)
- means \( \sim a_k \) commits itself to do \( l \)

- if \( l \) is a literal and controllable by \( a_k \), \( l \in \text{Eng}_-(a_k) \)
- means \( \sim a_k \) commits itself not to do \( l \)

The following table summarizes our modelization of sentences as \( \sim a_k \) commits itself to do \( l \).

<table>
<thead>
<tr>
<th>Sentence to modelize</th>
<th>Modelization</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_i ) commits itself to do ( l )</td>
<td>( l \in \text{Eng}_+(a_i) )</td>
</tr>
<tr>
<td>( a_i ) commits itself not to do ( l )</td>
<td>( l \in \text{Eng}_-(a_i) )</td>
</tr>
<tr>
<td>( a_i ) does not commit itself to do ( l )</td>
<td>( l \notin \text{Eng}_+(a_i) )</td>
</tr>
<tr>
<td>( a_i ) does not commit itself not to do ( l )</td>
<td>( l \notin \text{Eng}_-(a_i) )</td>
</tr>
</tbody>
</table>

We must impose consistency constraints to \( \text{Eng}_+ \) and \( \text{Eng}_- \): an agent \( a_k \) cannot commit itself to do \( l \) and \( \sim l \) and \( a_k \) cannot commit itself to do \( l \) and not to do \( l \).

**Constraint 1.** \( \forall a_k \in A \) \( \text{Eng}_+(a_k) \) is consistent and \( \text{Eng}_-(a_k) \) is consistent.

**Definition 15.** \( \text{Eng}_+(A) \) is the set of all agents positive commitments:

\[
\text{Eng}_+(A) = \bigcup_{a_k \in A} \text{Eng}_+(a_k)
\]

\( \text{Eng}_-(A) \) is the set of all agents “negative” commitments:

\[
\text{Eng}_-(A) = \{ l \in K B : \forall a_k \in A l \text{ controllable by } a_k \Rightarrow \sim l \in \text{Eng}_-(a_k) \}\]

\( \text{Eng}_-(A) \) means: if all the agents which control \( l \) commit themselves not to do \( l \) and if \( \sim l \in K B \), then \( \sim l \) will remain true. This supposes that there is no external intervention.

We emit an hypothesis about the consistency of commitments with the group’s CK goals:

**Hypothesis 1.**

\( \forall \varphi \sum \models I(\varphi|UI(K B)) \) and \( \varphi \) controllable by \( A \Rightarrow \text{Eng}_-(A) \cup \text{Eng}_+(A) \cup \{ \varphi \} \) is consistent

This hypothesis allows to avoid some problematic cases, for instance:

- the case where two agents which control \( l \) commit themselves one to do \( l \), the other to do \( \sim l \) (i.e. \( \text{Eng}_+(A) \) inconsistent)
- the case where there is a variable \( l \) so that all the agents that control \( l \) commit themselves not to do \( l \) and not to do \( \sim l \) (i.e. \( \text{Eng}_-(A) \) inconsistent);
- the case where the agents’ commitments contradict a group’s CK goal, for instance if \( l \) is a CK goal of the group and all the agents that control \( l \) commit themselves to do \( \sim l \).

In order to distribute goals to a group of agents, we must first verify the consistency of agents’ commitments with group’s CK goals. If the consistency is not verified, the agents must change their commitments.

4.3. Effective goals of an agent

If hypothesis 1 is verified, we are certain that the agents commitments are not contradictory with the group’s CK goal. We want now to determine the effective goals of each agent. Those goals are not CK goals, because, as we saw previously, each agent’s goals do not depend only on \( UI(K B) \) but also on the other agents’ commitments. It is intuitively correct to derive agent \( a_k \) goals from:

- the propositions which are uncontrollable by \( A \), i.e. \( UI(K B) \)
- the set of agents’ “positive” commitments, i.e. \( \text{Eng}_+(A) \)
- the set of agents’ “negative” commitments, i.e. \( \text{Eng}_-(A) \)

We note this set \( D(K B) \). \( D(K B) \) is used in the conditional part of the operator \( I(\sim|-) \) to derive effective goals.

**Definition 16.**

\[
D(K B) = UI(K B) \cup \text{Eng}_+(A) \cup \text{Eng}_-(A)
\]

The notion of effective goal of an agent is defined as follows:

**Definition 17.** Let \( \Sigma \) be a set of requirements and \( M_1, \ldots, M_n \) its models. Let \( K B \) be a knowledge base such that \( UI(K B) \) is complete.

For \( j \in \{1 \ldots n\} \), \( M_j^0 \) is the restriction of \( M_j \) to \( D(K B) \) worlds.

A proposition \( \alpha \) is an effective goal for \( a_k \) derived from \( \Sigma \) (we note \( E\text{Goal}_{a_k}(\alpha) \)) iff:
• ∀j ∈ {1...n} M_j^P = I(α)
• α is controllable by a_k

We use the I(−|−) operator to determine the effective goal, so we are certain that an agent cannot have contradictory effective goals.

We define also a notion of unsatisfied CK goal. If α is a CK goal for A, and if no agent has α’ for effective goal with ⊨ α’ → α, then α is unsatisfied.

**Definition 18.** Let α be a CK goal of A. α is unsatisfied iff:

\[ \bigcup_{α_k ∈ A} \{α’ : EGoal_{α_k}(α’}\} \not\subseteq α \]

We note it Nonsat(α).

**4.4. Example**

Let us resume the example of section 2.6. Let us consider a group of two agents A = \{a_1, a_2\}. We want to modelize the following requirements for A:

• there is a database
• if there is no database, there is no web page
• if there is a database, there is a web page

The set of requirements Σ is

Σ = \{E(db), E(¬wp|¬db), E(wp|db)\} whose only model is:

\[
\begin{array}{c}
\text{db} \quad \text{wp} \\
\text{¬db} \quad \text{¬wp} \\
\end{array}
\]

1. Suppose that KB = \{¬db, ¬wp\}, i.e. that the database and the web page are not yet created. Suppose that C(a_1) = \{db\} and C(a_2) = \{wp\}. In this case, the CK goal for A are db, db → wp and ¬db → ¬wp. Suppose that a_1 and a_2 do not commit themselves for anything. In this case, D(KB) = φ and the restricted model of Σ is:

\[
\begin{array}{c}
\text{db} \quad \text{wp} \\
\text{¬db} \quad \text{¬wp} \\
\end{array}
\]

The effective goals are EGoal_{a_1}(db) and EGoal_{a_2}(wp). a_1 has the task to create the database and a_2 has the task to create the web page.

2. Suppose that KB = \{¬db, ¬wp\}, C(a_1) = \{db\}, C(a_2) = \{wp\}. In this case, the CK goal for A are db, db → wp and ¬db → ¬wp. If a_1 does not commit itself for anything and a_2 commits itself not to do wp (for instance, a_2 has no time to create the web page), then Enq_{¬}(A) = \{¬wp\}, because a_2 is the only agent who can do wp and he decides not to do wp. So D(KB) = \{¬wp\} and the restricted model is:

\[
\begin{array}{c}
\text{db} \quad \text{¬wp} \\
\text{¬db} \quad \text{¬wp} \\
\end{array}
\]

The effective goals are EGoal_{a_1}(db) and EGoal_{a_2}(¬wp). db → wp is unsatisfied.

Notice that if a_1 commits itself to do db, then hypothesis 1. is falsified, because db → wp is a CK goal of the group \{a_1, a_2\}. The agents has to update their commitments.

3. Suppose that db is uncontrollable and KB = \{¬db, ¬wp\} (the database cannot be created, independently of the agents, for instance nobody knows how to create a database) and that wp is controllable by a_1 and a_2. Then the CK goal of \{a_1, a_2\} is ¬wp, i.e. the restricted model of Σ to calculate the CK goals is:

\[
\begin{array}{c}
\text{¬db} \quad \text{¬wp} \\
\text{¬db} \quad \text{¬wp} \\
\end{array}
\]

The agents cannot commit to do wp, because ¬wp is a CK goal of A. They have to “maintain” ¬wp. Notice that if wp ∈ KB in this case, the agents must delete the web page to fulfill the CK goal.

4. Suppose now that db is uncontrollable and KB = \{db, ¬wp\} (the database is already created for instance) and that wp is controllable by a_1 and a_2. Then the CK goal of \{a_1, a_2\} is wp. If a_1 does not commit itself to anything and if a_2 commits itself to do wp, the restricted model is:

\[
\begin{array}{c}
\text{db} \quad \text{wp} \\
\text{¬db} \quad \text{¬wp} \\
\end{array}
\]

In this case, both agents has to create the web page, because we can derive EGoal_{a_1}(wp) and EGoal_{a_2}(wp). We will discuss this case in the next section.
5. Conclusion

In this paper, we have proposed a formalism which allows one to represent requirements as orders on possible worlds and to determine from a set of requirements given a group of executing agents and a agency model for each agent:

- the controllable propositions by the group
- the CK goals of the group
- the effective goals of each executing agent

The first point we developed is the representation of requirements. According to us, a requirement cannot be only viewed as a most preferred world, but should also induce some less ideal cases. Dealing with such less ideal cases allows one to have a representation of alternatives that the participant in the RE process could use (for instance if a requirement is impossible to fulfill). We used the logic CO to express orders on worlds and two operators E(−) and E(−|−) to express requirements.

The second point we were interested in is the distribution of requirements among a group of executing agents. As we wrote in the introduction section, the distribution process can be used to verify requirements or simulate the building of the product concerned by the requirements. For instance, in example 4.4.2, the requirement db → wp is unsatisfied, because a2 commits itself not do wp. In this case, even if the requirements are consistent, correct... the project cannot be achieved because of one executing agent. Verifying that the creation of an “item” that will satisfy all the requirements is possible can be a new source for validation of requirements.

Let us notice that according to the definition of an effective goal, two executing agents may have the same tasks to realize. In example 4.4.4, a1 and a2 have both to create the web page (EGoala1(wp) and EGoala2(wp)). We can consider this case as “cooperation” between a1 and a2. But only a2 committed itself to do wp. Is it efficient to ask a1 for doing wp? This optimization problem can be solved by using different “efficiency” criterions. Suppose that a1 had a lot of other effective goals (a very loud task charge), then it would be judicious to remove wp from a2’s effective goals. We could also introduce a preference order on agents: a1 could be preferred to a2 to do wp, because a1 is cheaper, or faster for instance. In this case, a2 should let a1 do wp. In the light of such examples, working on defining a strategy set Strat(aK) for each agent would be interesting. Strat(aK) could be based on an order on the executing agents for each CK goal of the group. This order can reflect various preference relations: an agent can be preferred to another one because it is more competent to execute a task, because it consumes less resources etc.

Thus, we would be able to define particular distribution strategies for a group of agents and find the one fittest for a specific problem (cooperation, taking the best agent for each task...). The most interesting work is to obtain postulates about Strat that permit to qualify the global strategy given Strat(aK) for each agent aK.

References