

# Bridging Core Ontologies of Requirements and Norms towards Automated Engineering of Norm-Governed Multiagent Systems

Ivan J. JURETA <sup>a</sup>, Martin KOLLINGBAUM <sup>b</sup>, Katia SYCARA <sup>b</sup>, and  
Stéphane FAULKNER <sup>a</sup>

<sup>a</sup> *PreCISE, University of Namur, iju@info.fundp.ac.be, sfaulkne@fundp.ac.be*

<sup>b</sup> *Robotics Institute, Carnegie Mellon University, {mkolling, katia}@cs.cmu.edu*

**Abstract.** A software agent can act within a norm-governed multiagent system only if it accepts to behave in accordance with the norms – i.e., obligations, permissions, and prohibitions – governing that system. Norms ensure that the participating agents behave only in ways that lead the multiagent system as a whole to satisfy the requirements – i.e., goals, softgoals, quality constraints, preferences, and so on – set by the stakeholders of that system. Our aim is to facilitate the engineering of norm-governed multiagent systems by automating part of the effort involved in the specification of norms. Current practice is to specify requirements and then specify norms later on in the engineering process. We suggest instead that specifications of norms be derived automatically from specifications of requirements. To do so, we outline a core ontology of requirements and a core ontology of norms and map the former to the latter by way of the DOLCE foundational ontology. We show how these mappings allow us to define algorithms for the automatic definition of norms from a specification of requirements.

**Keywords.** norms, requirements, multiagent systems, ontology mapping, distributed description logics

*Started on February 22, 2008. Last change: March 17, 2008; first complete draft; author list open; feedback welcome; the first author is responsible for errors.*

## Introduction

Successful response to the increasing complexity in the engineering and management of modern information systems requires the use of multiagent systems (MAS's) [32,30], in which potentially many software agents act and interact in order to achieve individual and joint goals. A key problem with a MAS is the coordination of agents towards the achievement of joint goals (i.e., goals of the entire MAS); indeed, the agents are heterogeneous, distributed over the computing resources of various providers, and act on behalf and in the interest of these providers, being therefore prone to behave in ways that, while beneficial to their individual aims, may be harmful to the purpose of the entire MAS. It is increasingly argued that coordination in a MAS requires the definition of *norms*, which, broadly speaking, act as constraints on the behavior of individual agents [57,52,63]. In

order to participate in a *norm-governed MAS* – i.e., a MAS in which norms constrain admissible behaviors – an agent must accept the norms established for that system.

Two important tasks in the process of engineering norm-governed MAS are requirements engineering and norm engineering. The former is the first step of norm-governed MAS engineering, during which the stakeholders of the MAS (including, among others, its owners and users) communicate to the MAS engineer the requirements about how and how well the MAS should behave and what it should produce when deployed. The result of requirements engineering is a (formal) requirements specification. Norm engineering comes later on in the norm-governed MAS engineering process (e.g., [67]) and results in a norm specification, which indicates the obligations, permissions, and prohibitions, to which the agents participating in the norm-governed MAS must commit. At present, norm engineering is largely separated from requirements engineering (e.g., [52,67,63]).

That requirements and norm engineering are separate may not appear surprising, since each seemingly relies on different conceptualizations: the ontology instantiated when specifying requirements is different from that instantiated when specifying norms. The former features concepts such as, e.g., *goal*, *softgoal*, *domain assumption*, while the latter incorporates, e.g., the notions of *obligation*, *permission*, and *prohibition*.

Although conceptualizations of requirements and norms differ, it is clear that norms must be defined in such a way that commitment thereto guarantees that the norm-governed MAS fulfills its requirements. In other words, precise relationships must exist between norms and requirements, because the purpose of norms is to ensure that agents behave so as to satisfy the requirements. For instance, when facing a requirement for a banking application engineered on principles of a norm-governed MAS, that a Letter of Credit cannot be issued is a deposit is not recorded, knowing the relationships between requirements and norms would allow us to automatically define norms to ensure that this requirement is satisfied at runtime. Knowledge of these relationships would allow us to automate a significant part of norm engineering, as the specification of norms could be automatically derived from the specification of requirements.

The primary contribution of this paper are mappings between a core ontology of requirements and a core ontology of norms. The second contribution of the paper is an example of how the mappings serve in the definition of algorithms for the automatic specification of norms from a specification of requirements. The paper thereby provides a conceptual basis for the automation of the norm engineering task in the norm-governed MAS engineering process. We focus on core ontologies for requirements and for norms, that is, ontologies featuring minimal sets of concepts needed in the specification of, respectively, requirements and norms. Among the various ways to define mappings between ontologies (see, e.g., [39] for an overview), we adopt the following overall approach, reflected in the structure of the paper: (1) we introduce a core ontology of requirements (denoted OR herein) and a core ontology of norms (denoted ON) (Section 1); (2) we choose a foundational ontology (here, DOLCE [44]) that is grounded in ontological choices attractive for the present discussion, and establish how each of the ontologies – resp., OR and ON – maps individually to the foundational ontology of choice, which allows us to define how OR maps to ON (Section 2). To show the benefit of the mappings, Section 3 gives an example of an algorithm for automatically deriving a fragment of a specification of norms from a fragment of a specification of requirements for a banking application engineered on the principles of norm-governed MAS. Section 4 discusses related efforts,

and Section 5 closes the paper with a summary of contributions and directions for future work.

## 1. Ontologies of Requirements and Norms

### 1.1. OR: Core Ontology of Requirements

The requirements engineering process is one in which the stakeholders communicate to the software engineer the requirements about how and how well the future system is expected to operate and what it is to produce once deployed. Utterances that the stakeholders make in communicating with the engineer are actions intended to advance stakeholders' personal desires, intentions, and beliefs, in the aim of ensuring that the engineer can produce a specification of requirements that then leads to the development of the system responsive to the communicated concerns. The scope of the core ontology of requirements is therefore limited to concepts and relationships that allow the representation of all concerns communicated by way of speech acts. It is depending on the illocutionary force that the software engineer will write the content of the communication as instances of one or another concept in the core ontology of requirements. The ontology therefore covers content conveyed by [55,56]: (i) *assertives*, which assert the proposition that the speaker believes is true; (ii) *directives*, which convey the proposition that the speaker wants to see become true; (iii) *commissives* stating what the speaker intends to (do to) make a proposition true; (iv) *expressives* that convey a speaker's emotion/attitude about herself or the hearer; (v) *declarations* which by the very act of being stated make a proposition true; and (vi) *representative declaratives*, which recognize the truth of a proposition that has been made true by a declaration. We know from Searle [55,56] that assertives and representative declaratives communicate beliefs, directives desires, commissives intentions, expressives attitudes, declarations become beliefs when communicated. The term *attitude* here is attributed the meaning given by psychologists: an attitude is identified most closely with affect, i.e., a general evaluative reaction (e.g., "I like Y" or "I like Y more than Z") [1].<sup>1</sup>

Our ontology of requirements is specified in Table 1 using OWL DL. The meaning intended for the concepts in the ontology is given in Table 2 and arises from the purpose for which the ontology is defined. The purpose of OR is to serve methodologies for requirements engineering, that is, to guide elicitation and organization of requirements-related information. Therefore, content of what the stakeholders communicate is distinguished from the illocutionary force used in communication. As said above, the software engineer classifies some communicated content according to the definitions given in Table 2: facing, e.g., a statement such as that in Ex.1 in Table 2, communicated by way of an assertive speech act, the engineer will consider it to be an instance of a domain assumption, whereas the statement Ex.2, communicated by way of a directive speech act and referring to an observable quality (i.e., number of screens needed to fill out an online request for a Letter of Credit) of the online process for requesting a Letter of Credit, will be considered an instance of a quality constraint. The methodological value of OR lies

---

<sup>1</sup>This use of the term 'attitude' here is not to be confused with its common use in philosophy, where beliefs, desires, and intentions are called propositional attitudes [51]. Expressives in Searle's classification convey what psychologists call "attitudes".

**Table 1.** Specification of the core ontology of requirements (OR).

1:	REQ-STATEMENT	$\equiv$	DOMAIN-ASSUMPTION $\sqcup$ QUALITY-CONSTRAINT $\sqcup$ GOAL $\sqcup$ SOFTGOAL $\sqcup$ PLAN
2:		$\perp \sqsubseteq$	DOMAIN-ASSUMPTION $\sqcap$ QUALITY-CONSTRAINT $\sqcap$ GOAL $\sqcap$ SOFTGOAL $\sqcap$ PLAN
3:	REQ-STATEMENT	$\equiv$	OPTIONAL-REQ-STATEMENT $\sqcup$ COMPULSORY-REQ-STATEMENT
4:		$\perp \sqsubseteq$	OPTIONAL-REQ-STATEMENT $\sqcap$ COMPULSORY-REQ-STATEMENT
5:	refined-by	$\equiv$	refine <sup>-1</sup>
6:	refine	$\equiv$	refined-by <sup>-1</sup>
7:		$\top \sqsubseteq$	$\forall$ refine.REQ-STATEMENT
8:		$\top \sqsubseteq$	$\forall$ refined-by.REQ-STATEMENT
9:	GOAL	$\sqsubseteq$	$\exists$ satisfied-by.PLAN
10:	QUALITY-CONSTRAINT	$\sqsubseteq$	$\exists$ satisfied-by.PLAN
11:	satisfied-by	$\equiv$	satisfy <sup>-1</sup>
12:	satisfy	$\equiv$	satisfied-by <sup>-1</sup>
13:		$\top \sqsubseteq$	$\forall$ satisfy.GOAL $\sqcup$ $\forall$ satisfy.QUALITY-CONSTRAINT
14:		$\top \sqsubseteq$	$\forall$ satisfied-by.PLAN
15:	SOFTGOAL	$\sqsubseteq$	$\exists$ approximated-by.QUALITY-CONSTRAINT
16:	approximated-by	$\equiv$	approximate <sup>-1</sup>
17:	approximate	$\equiv$	approximated-by <sup>-1</sup>
18:		$\top \sqsubseteq$	$\forall$ approximate.SOFTGOAL
19:		$\top \sqsubseteq$	$\forall$ approximated-by.QUALITY-CONSTRAINT
20:	ATTITUDE	$\equiv$	OPTIONAL-REQ-STATEMENT $\sqcup$ COMPULSORY-REQ-STATEMENT $\sqcup$ (PREF-COMPARED $\sqcap$ $\exists$ preferentially-comparable.PREF-COMPARED)
21:	PREF-COMPARED	$\sqsubseteq$	REQ-STATEMENT $\sqcup$ ATTITUDE

in the fact that techniques exist in requirements engineering for the analysis of each of these concepts and the use of its instances in subsequent steps of the software development process.

To help the reader understand the basics of the ontology of requirements and its use, the rationale for the ontology is explained below by referring to both Tables 1 and 2. Beliefs communicated by way of assertive, declarative, or representative declarative speech acts constrain the possible states of the world only to those in which beliefs are not violated. They are captured by DOMAIN-ASSUMPTION, which should not be violated by the system or its relevant surroundings. The concepts of GOAL and QUALITY-CONSTRAINT are introduced to cover the classical kinds of “requirement” – namely, the functional and nonfunctional requirements. It is widely accepted in requirements engineering that functional requirements refer to what the system does, as opposed to how well the system does what it does (e.g., [60]). In Table 2, the example for content classified as a quality constraint indicates how well the process of requesting a Letter of Credit performs for it identifies a measure on the behavior of the system (i.e., the number of different screens the user needs to go through to complete a request for a Letter of Credit). In other words, that statement characterizes the requesting process – it points to the existence of “qualities” for which only some (sets of) possible values are desired. The functional vs. nonfunctional distinction is grounded in the notion of “quality”, whereby

**Table 1.** (Continued.) Specification of the core ontology of requirements (OR).

22:	preferentially-strictly-better	$\equiv$	preferentially-strictly-worse <sup>-1</sup>
23:	preferentially-equivalent-or-worse	$\equiv$	preferentially-equivalent-or-better <sup>-1</sup>
24:	preferentially-strictly-worse	$\equiv$	preferentially-strictly-better <sup>-1</sup>
25:	preferentially-equivalent-or-better	$\equiv$	preferentially-equivalent-or-worse <sup>-1</sup>
26:	preferentially-strictly-equivalent	$\equiv$	preferentially-strictly-equivalent <sup>-1</sup>
27:	preferentially-not-equivalent	$\equiv$	preferentially-not-equivalent <sup>-1</sup>
28:	preferentially-strictly-better	$\sqsubseteq$	preferentially-equivalent-or-better
29:	preferentially-strictly-better	$\sqsubseteq$	preferentially-not-equivalent
30:	preferentially-strictly-worse	$\sqsubseteq$	preferentially-comparable
31:	preferentially-strictly-worse	$\sqsubseteq$	preferentially-comparable-or-worse
32:	preferentially-strictly-worse	$\sqsubseteq$	preferentially-not-equivalent
33:	preferentially-equivalent-or-better	$\sqsubseteq$	preferentially-comparable
34:	preferentially-strictly-equivalent	$\sqsubseteq$	preferentially-equivalent-or-better
35:	preferentially-strictly-equivalent	$\sqsubseteq$	preferentially-equivalent-or-worse
36:	preferentially-not-equivalent	$\sqsubseteq$	preferentially-comparable
37:		T	$\sqsubseteq \forall$ preferentially-strictly-better <sup>-1</sup> .PREF-COMPARED
38:		T	$\sqsubseteq \forall$ preferentially-comparable <sup>-1</sup> .PREF-COMPARED
39:		T	$\sqsubseteq \forall$ preferentially-equivalent-or-worse <sup>-1</sup> .PREF-COMPARED
40:		T	$\sqsubseteq \forall$ preferentially-strictly-worse <sup>-1</sup> .PREF-COMPARED
41:		T	$\sqsubseteq \forall$ preferentially-strictly-equivalent <sup>-1</sup> .PREF-COMPARED
42:	preferentially-strictly-equivalent	$\equiv$	preferentially-strictly-equivalent <sup>-1</sup>
43:	preferentially-not-equivalent	$\equiv$	preferentially-not-equivalent <sup>-1</sup>
44:		T	$\sqsubseteq \forall$ preferentially-strictly-better.PREF-COMPARED
45:		T	$\sqsubseteq \forall$ preferentially-comparable.PREF-COMPARED
46:		T	$\sqsubseteq \forall$ preferentially-strictly-worse.PREF-COMPARED
47:		T	$\sqsubseteq \forall$ preferentially-equivalent-or-better.PREF-COMPARED
48:		T	$\sqsubseteq \forall$ strictly-equivalent.PREF-COMPARED
49:			TRANSITIVE(preferentially-strictly-better)
50:			TRANSITIVE(preferentially-strictly-worse)
	51:		TRANSITIVE(preferentially-equivalent-or-better)
	52:		TRANSITIVE(preferentially-strictly-equivalent)

“quality” is understood as in DOLCE [44]: a quality is a basic entity that we can perceive or measure and is of some quality type (e.g., color, size). A quality is distinguished from a quality value, whereby the value describes the position of an individual quality within a certain quality space (which is a conceptual space, in the sense of Gärdenfors [26]). A requirements statement that describes qualities and constrains quality values is taken to be a nonfunctional requirement; otherwise it is a functional requirement. In requirements engineering, nonfunctional requirements are taken to include considerations such as security, safety, maintainability, convenience, and so on. Given that a quality is a perceivable and measurable entity that inheres in other entities, there must be a quality space and values for, e.g., convenience in order to call it a quality, and thereby state that asking for, e.g., “high convenience” (as in the example for the softgoal concept in Table 2) is a quality constraint. For all practical purposes, qualities take the form of measures for which we make explicit the desired values, that is, we define quality constraints for the system to satisfy. Hence, quality constraints cover some of nonfunctional requirements

**Table 2.** Meaning intended for concepts in the core ontology of requirements (OR). Each definition is immediately followed by an example, that is, a statement in natural language that is classified as an instance of the concept or relationship.

<p>DOMAIN-ASSUMPTION: Believed content communicated by way of assertive, declarative, or representative declarative speech acts is a domain assumption.</p> <p>(Ex.1) “The Banker cannot issue the Letter of Credit if no corresponding deposit has been recorded.”</p>
<p>QUALITY-CONSTRAINT: Desired content communicated by way of a directive speech act is a quality constraint if and only if the content in question constrains quality values. Described qualities must have quality space with a well-defined and shared structure.</p> <p>(Ex.2) “It should be possible to fully fill out an online request for a Letter of Credit through less than 5 different screens.”</p>
<p>GOAL: Desired content communicated by way of a directive speech act is a goal if and only if the content in question neither describes qualities nor constrains quality values.</p> <p>(Ex.3) “Issue a letter of credit immediately after the deposit is recorded and the credit capacity of the requester verified.”</p>
<p>SOFTGOAL: Desired content communicated by way of a directive speech act is a softgoal if and only if the content describes qualities or constrains quality values, whereby the described qualities must have a quality space with a subjective and/or ill-defined structure.</p> <p>(Ex.4) “The process of requesting and obtaining the Letter of Credit should be convenient for the requester.”</p>
<p>PLAN: Intended content communicated by way of a commissive speech act is a plan.</p> <p>(Ex.5) “I will ensure that the letter of credit is issued immediately after the deposit is recorded.”</p>
<p>ATTITUDE: Attitudinal content communicated by way of an expressive speech act is an attitude if and only if it evaluates in terms of favor or disfavor one or more instances of domain assumptions, quality constraints, goals, softgoals, or plans.</p> <p>(Ex.6) “It is preferred that the letter of credit be issued not on the basis of the received deposit only, but after the deposit has been recorded and the credit capacity of the customer verified.”</p>

but not all: nonfunctional requirements on, e.g., convenience, security, maintainability, and so on, are not quality constraints – we call them softgoals instead, although they still rely on the concept of quality (as in DOLCE). The SOFTGOAL differs from the quality constraint in that the quality space has a subjective and/or ill-defined structure (e.g., if person 1 evaluates convenience differently – that is, over different criteria – than person 2 does, then there is a difference in the structure of the underlying quality space that each of the two refers to, which leads us to call that quality space subjective in Table 2).

The content of expressive speech acts gives attitudes. An instance of ATTITUDE amounts to a description of an evaluation in terms of degree of favor or disfavor [21]. Such degrees vary in sign (positive or negative) and in intensity, whereby the intensity of the valuation is relative: considering an object of attitude on its own involves implicit comparison to a set of objects perceived by the evaluator to be of the same kind [38]. As Kahneman and colleagues observe [38], “objects of attitudes [as the term is used in psychology] include anything that people can like or dislike, wish to protect or to harm, to acquire or to reject”. Stakeholders can thus evaluate favorably or disfavorably, and with different intensity individual or alternative domain assumptions, plans, goals, quality constraints, or softgoals.

It is important to understand the role of attitudes in the ontology of requirements. The information that instances of the attitude concept convey gives us either an evaluation of a single other statement of requirements, or a comparative evaluation of two or

more requirements statements, as reflected in the specification in Table 1. If we have an evaluation of a single requirements statement, we interpret it as saying whether the content of that statement is compulsory or optional; if compulsory, then the future system must obey that statement (e.g., if it is a goal, the system must have behaviors that satisfy that goal, if it is a domain assumption, the system must not behave in ways that violate the domain assumption). If we have a comparative evaluation of two or more statements of requirements, we interpret it as a preference order over those statements. If we find out from such an attitude that satisfying some goal X is evaluated as more desirable than doing so for another goal Y, we understand this as saying that X is preferred to Y. How requirements statements can relate in terms of preference is given in Table 1, and is a usual conceptualization of preference orders: say that we relate two requirements statements,  $R_1$  and  $R_2$ , then: (i)  $R_1$  may be strictly preferred to  $R_2$  (**preferentially-strictly-better** or **preferentially-strictly-worse**), (ii)  $R_1$  may be at least as preferred as  $R_2$  (**preferentially-equivalent-or-better** or **preferentially-equivalent-or-worse**), (iii)  $R_1$  may be equally preferred to  $R_2$  (**preferentially-strictly-equivalent**). Further reasons are given elsewhere [37] as to why – in light of seminal works in requirements engineering – the ontology outlined in Tables 1 and 2 is appropriate as a core ontology of requirements, and that it carries only the basic concepts and relationships necessary to the software engineer in resolving the requirements problem, which defines what it means for the requirements engineering effort to be successfully completed.

The specification in Table 1 carries refinement, satisfaction, and approximation relationships. The meaning for these becomes clear from the use of the ontology. Given the ontology of requirements, requirements engineering starts off with the elicitation from stakeholders and documentation of natural language statements about the future system, then proceeds to the classification of these as instances of the concepts and relationships of the ontology. Analysis of the requirements – by way of available techniques in the field of requirements engineering for modeling (e.g., [47,17,11]), refinement (e.g., [18]), verification (e.g., [24]), negotiation (e.g., [2]), conflict resolution [61], and so on – results in a formal specification of requirements agreeable to the stakeholders. In that specification, each instance of SOFTGOAL will be approximated by verifiable instances of quality constraints (hence **approximated-by** and **approximate**), so that satisfying the quality constraints is interpreted as satisfying the softgoal that these quality constraints approximate. An instance of a requirements statement can be refined by other instances of the requirements statement of the same kind, in order to render the requirements more precise (**refined-by** and **refine**).

In a requirements specification, the instances of goal, quality constraint, domain assumption, plan, and preferences (that arise from attitudes) are written in a structured, preferably formal notation, such as, e.g., Horn clauses or formulas of some first order logic. For example, if we have a goal “Record a deposit in electronic and paper form if the deposit is received and approved”, then we can specify it using Horn clauses with:  $eL : elLog, pL : paLog (logs(eL,pL,d) \leftarrow isPaperLog(pL,d), isElectronicLog(eL,d))$ , where  $eL, pL, d$  are instances of concepts defined in an ontology for the banking domain.<sup>2</sup> The reader will note at this point

<sup>2</sup>It is unimportant for the present discussion what formal language is used; we merely wish to point out to the reader new to requirements engineering that some formal language will be used in the requirements specification. This is necessary, as shown later on, if we are to derive automatically executable norms from requirements.

that our aim of automatically deriving norms from requirements translates into determining how to carry each fragment of a specification (such as that just given) from the formal specification of requirements into a specification of norms. This leads us to the ontology for norms, discussed below. Note that, since we approximate softgoals by quality constraints, it is enough to define norms that satisfy quality constraints in order to ensure that softgoals are satisfied. Hence, softgoals do not need mappings to norms.

### 1.2. ON: Core Ontology of Norms (for Norm-Governed MAS)

An ontology of norms falls by definition within the legal domain. Theories of law usually contain ontological assumptions, whereby these assumptions are determined by some specific theoretical aim which lies beyond the ontology itself. For example, as Breuker and Winkels observe [7], Hohfeld's [28] aim was to study legal competences and legal responsibilities, which led him to manipulate notions of right, duty, no-right, privilege, power, and so on. Valente and colleagues [59] and Breuker and colleagues [5] suggest core ontologies of law in the aim of assisting knowledge engineering and information management in relation to the legal domain (for similar efforts, see, e.g., [9]). The aim here is less ambitious and has a considerably more restricted scope: as outlined earlier, we are interested in automating the definition of norms that can be interpreted by software agents in a norm-governed MAS. Therefore, not only must the ontology of norms be (1) appropriate with regards to the legal domain and norms in particular, but it must also be (2) expressive only to the extent that it remains compatible with reasoning model of the software agent that is to interpret and obey the norms.

To ensure that our first criterion is satisfied – i.e., (1) that ON is appropriate with regards to the legal domain and norms in particular – we considered available legal ontologies, and in particular, the OWL Ontology of Basic Legal Concepts from Breuker and colleagues [6] and the Core Legal Ontology from Gangemi and colleagues [25].

- In the norm module from the OWL Ontology of Basic Legal Concepts from Breuker and colleagues [6], a NORM applies to a situation, that is, it makes a situation NORMATIVELY-QUALIFIED. The norm can allow, making ALLOWED the normatively qualified situation. If OBLIGED, the situation is allowed and it is assumed that there is commitment to bring about or maintain the situation. The OBLIGATION and PROHIBITION completely partition NORM, for obligation and prohibition are simply different ways to put the same thing into words. PERMISSION differs in that it allows but does not prohibit.
- The Core Legal Ontology (CLO) organizes legal concepts and relations based on formal properties defined in DOLCE+ (DOLCE extended with the “descriptions and situations” ontology [25]). In CLO, “the legal world is conceived as a description of social reality, an ideal view of the behaviour of a social group, according to a system of rules that are commonly accepted and acknowledged.” Although the OWL Ontology of Basic Legal Concepts differs from CLO at the upper levels, it does feature similar notions at the lower levels.<sup>3</sup> In CLO, a legal description  $D_T$  is the content of a norm, and is assumed to be the reification of a theory  $T$  that

<sup>3</sup>An important difference is how the content of a norm is conceptualized: as the content refers to the real world, it depends strongly on what categories are taken to exist in the real world, which is where the main differences lie between DOLCE and the foundational ontology subsumed in the OWL Ontology of Basic Legal Concepts [27].

**Table 3.** Specification of the core ontology of norms for norm-governed MAS (ON) that follows the norm module of the OWL Ontology of Basic Legal Concepts.

53:	NORM	≡	∃qualifies.NORMATIVELY-QUALIFIED
54:	NORM	⊆	Qualification
55:	PERMISSION	⊆	NORM
56:	PERMISSION	≡	∃allowed.ALLOWED ⊓ ∃allowed.ALLOWED
57:	PROHIBITION	⊆	PERMISSION
58:	PROHIBITION	≡	∃allows.OBLIGED ⊓ ∃allows.OBLIGED ⊓ ∃disallows.DISALLOWED ⊓ ∃disallows.DISALLOWED
59:	OBLIGATION	≡	PROHIBITION
60:	NORMATIVELY-QUALIFIED	≡	∃qualified-by.NORM
61:	ALLOWED	⊆	NORMATIVELY-QUALIFIED
62:	ALLOWED	≡	∃allowed-by.PERMISSION
63:	OBLIGED	⊆	ALLOWED
64:	DISALLOWED	⊆	NORMATIVELY-QUALIFIED
65:	DISALLOWED	≡	∃disallowed-by.PROHIBITION
66:	STRICTLY-ALLOWED	⊆	ALLOWED
67:	STRICTLY-DISALLOWED	⊆	DISALLOWED
68:	ALLOWED-AND-DISALLOWED	≡	DISALLOWED ⊓ ALLOWED
69:	disallows	⊆	qualifies
70:	allows	⊆	qualifies
71:	allowed-by	≡	allows <sup>-1</sup>
72:	disallowed-by	≡	disallows <sup>-1</sup>
73:	NORMATIVELY-QUALIFIED	≡	∃normatively-comparable.NORMATIVELY-QUALIFIED
74:	ALLOWED	≡	∃normatively-equivalent-or-worse.NORMATIVELY-QUALIFIED
75:	OBLIGED	≡	∃normatively-strictly-worse.DISALLOWED
76:	DISALLOWED	≡	∃normatively-strictly-better.ALLOWED
77:	normatively-strictly-better	≡	normatively-strictly-worse <sup>-1</sup>
78:	normatively-equivalent-or-worse	≡	normatively-equivalent-or-better <sup>-1</sup>
79:	normatively-strictly-worse	≡	normatively-strictly-better <sup>-1</sup>
80:	normatively-equivalent-or-better	≡	normatively-equivalent-or-worse <sup>-1</sup>
81:	normatively-strictly-equivalent	≡	normatively-strictly-equivalent <sup>-1</sup>
82:	normatively-not-equivalent	≡	normatively-not-equivalent <sup>-1</sup>
83:	normatively-strictly-better	⊆	normatively-equivalent-or-better
84:	normatively-strictly-better	⊆	normatively-not-equivalent
85:	normatively-strictly-worse	⊆	normatively-comparable

(potentially) formalizes the content of a norm or a bundle of norms, while a legal case  $C_S$  is assumed to be a state of affairs that is a logical model of the theory  $T$ . Hence, in CLO, if the aim is to assess some course of events with regards to a norm (to see if how the events occurred satisfies the norm), we would see the content of the norm as a description of a course of events that is acceptable, and given the actual events, we would compare the norm to them in order to assess their conformity to the norm. CLO is not explicit on the notions of prohibition, obligation, and permission, but they can be defined within CLO.

Choosing either of the two ontologies above satisfies our first criterion, so that the choice between the OWL Ontology of Basic Legal Concepts and CLO is guided by the

**Table 3.** (Continued.) Specification of the core ontology of norms for norm-governed MAS (ON) that follows the norm module of the OWL Ontology of Basic Legal Concepts.

86:	normatively-strictly-worse	$\sqsubseteq$	normatively-comparable-or-worse
87:	normatively-strictly-worse	$\sqsubseteq$	normatively-not-equivalent
88:	normatively-equivalent-or-better	$\sqsubseteq$	normatively-comparable
89:	normatively-strictly-equivalent	$\sqsubseteq$	normatively-equivalent-or-better
90:	normatively-strictly-equivalent	$\sqsubseteq$	normatively-equivalent-or-worse
91:	normatively-not-equivalent	$\sqsubseteq$	normatively-comparable
92:		T $\sqsubseteq$	$\forall$ normatively-strictly-better <sup>-1</sup> .DISALLOWED
93:		T $\sqsubseteq$	$\forall$ normatively-comparable <sup>-1</sup> .NORMATIVELY-QUALIFIED
94:		T $\sqsubseteq$	$\forall$ normatively-equivalent-or-worse <sup>-1</sup> .ALLOWED
95:		T $\sqsubseteq$	$\forall$ normatively-strictly-worse <sup>-1</sup> .OBLIGED
96:		T $\sqsubseteq$	$\forall$ normatively-strictly-equivalent <sup>-1</sup> .ALLOWED
97:	normatively-strictly-equivalent	$\equiv$	normatively-strictly-equivalent <sup>-1</sup>
98:	normatively-not-equivalent	$\equiv$	normatively-not-equivalent <sup>-1</sup>
99:		T $\sqsubseteq$	$\forall$ normatively-strictly-better.OBLIGED
100:		T $\sqsubseteq$	$\forall$ normatively-comparable.NORMATIVELY-QUALIFIED
101:		T $\sqsubseteq$	$\forall$ normatively-strictly-worse.DISALLOWED
	102:	T $\sqsubseteq$	$\forall$ normatively-equivalent-or-better.ALLOWED
	103:	T $\sqsubseteq$	$\forall$ strictly-equivalent.ALLOWED
	104:		TRANSITIVE(normatively-strictly-better)
	105:		TRANSITIVE(normatively-strictly-worse)
	106:		TRANSITIVE(normatively-equivalent-or-better)
	107:		TRANSITIVE(normatively-strictly-equivalent)

purpose of the norm ontology in this paper. Because our aim is to facilitate the specification of norms for norm-governed MAS, we are constrained to choose an ontology that fits the reasoning model of a norm-governed software agent. This leads us to consider in more detail the second criterion.

To satisfy our second criterion – i.e., (2) that ON is expressive only to the extent that it remains compatible with reasoning model of the software agent that is to interpret and obey the norms – it is relevant to briefly explain the reasoning abilities of the software agent in a norm-governed MAS so as to understand how norms are interpreted and govern behavior. A software agent that is acting in accordance to norms must be able to adopt norms such as obligations, permissions, and prohibitions as they are established within a norm-governed MAS. The agent must be able to process them correctly, and anticipate the potential interactions between the effects of its actions and its norms. Kollingbaum’s [41,42,62] norm-governed practical reasoning software agent satisfies these expectations; it maintains three basic sets of objects: a set of beliefs, a set of plans and a set of norms. The set of beliefs reflects the agent’s current state and its perception of the world. The set of plans represents the capabilities of the agent. The set of norms contains obligations, permissions and prohibitions adopted from norm-governed MAS. Depending on the beliefs, formed also through the acquisition of norms and the past execution of plans, the agent instantiates current norms and available plans, compares candidate plans and selects a plan that is consistent with the norms, which then leads it to execute the chosen plan and update its beliefs. Triggered by some event (e.g., the arrival of a new norm), the agent engages in this same reasoning cycle before choos-

**Table 4.** Meaning intended for concepts in our core ontology of norms. Intended meaning is given according to how obligations, permissions, and prohibitions affect the behavior of the software agent in a norm-governed MAS.

<p><b>OBLIGATION:</b> An obligation motivates a software agent to act in order to bring about conditions that the obligation refers to (called: allowed conditions), and blocks the software agent from bringing about those conditions (called: disallowed conditions) that violate the allowed conditions. Allowed and disallowed conditions are complete partitions of all conditions referred to by the obligation.</p>
<p><b>PERMISSION:</b> A permission does not block a software agent from acting in order to bring about conditions that the permission refers to (called: allowed conditions), and does not block the agent from acting in order to bring about conditions (called: disallowed conditions) that violate allowed conditions. Allowed and disallowed conditions are complete partitions of all conditions referred to by the permission.</p>
<p><b>PROHIBITION:</b> A prohibition blocks the software agent from bringing about conditions that the prohibition refers to (called: disallowed conditions), and motivates the software agent to act in order to bring about conditions (called: allowed conditions) that violate the disallowed conditions. Allowed and disallowed conditions are complete partitions of all conditions referred to by the prohibition.</p>

ing how to act. A norm in a norm-governed MAS is thus either an obligation, a prohibition, or a permission.<sup>4</sup> Obligations motivate the agent to either achieve a state of affairs or to perform a specific action or bring about a state of affairs. Based on such a motivation, a norm-governed agent will select an appropriate plan from its current behavioural repertoire. Prohibitions require the agent not to achieve a state of affairs or perform an action – the agent is forbidden to pursue a specific activity. Prohibitions explicitly restrict the choices of activities the agent can ideally employ. Permissions explicitly allow the achievement of a state of affairs or the performance of an action. This conceptualization of norm is common in norm-governed MAS that follow standard ideas in law. Namely, a norm combines two meanings: it is deontic, saying what is acceptable, and it commits the agent to act in the acceptable way (or bring about the acceptable state). Following the desiderata for deontic knowledge representation, the conceptualization of norm ensures that what is obligatory is permitted, and that the impossible is not obligatory, and – by techniques for the resolution of conflict between norms [62] – that obligations are not conflicting.

The specification of ON is shown in Table 3. The meaning intended for the concepts is given in Table 4. Axioms from the OWL Ontology of Basic Legal Concepts are reused, while the intended meaning is that applicable to the problem domain and discussion of this paper. To satisfy our first criterion, the general characteristics of norms, obligations, permissions, and prohibitions are adopted from the OWL Ontology of Basic Legal Concepts (hence the reuse of axioms), as shown in Table 3. These characteristics are (i) the relationships between norm, obligation, permission, and prohibition concepts, (ii) the properties of these concepts that arise from the fact that the latter three are kinds of norm, and that the norm combines two performative meanings: it is deontic in that it states acceptability of something, and it assumes commitment to bring about what is acceptable. Since both the OWL Ontology of Basic Legal Concepts and CLO admit these characteristics for the norm concept, our core ontology satisfies the first criterion. Our second criterion is satisfied as well, since the norm and its relationships to obligation, permission,

<sup>4</sup>The reader interested in further details of these models will refer to [42,62,52,63], which include methods on the automated detection and resolution of inconsistencies between norms.

and prohibition described in Table 3 fit to the conceptualization of norm adopted by the norm-governed software agent. What differs in the notion of norm here from that in the OWL Ontology of Basic Legal Concepts or CLO is the intended meaning. The intended meaning given in the present paper is chosen to fit the purpose of the norm concept: the ontology of norms herein serves – just as the ontology of requirements described earlier – a methodological purpose. That is, just as the goal concept from the ontology of requirements is instantiated to characterize the content of some communication from the stakeholder, the obligation concept is instantiated to characterize some action that the agent is obliged to execute or a state of affairs that the agent is obliged to bring about. Defining our ontology of norms in such a way is uncontroversial – as Breuker and colleagues’ observe [6]: “Ontologies contain definitions of terms, but these definitions are not independent of the context of use.” We admit the general characteristics necessary for something to be called a norm, an obligation, a permission, or a prohibition, and the meaning intended herein acknowledges this, but we also adopt the specific meaning of these concepts – we specialize them – to reflect the context in which they are to fulfil their purpose, namely, the engineering of norm-governed MAS.

## 2. From OR to ON by way of DOLCE

### 2.1. Preliminaries: DOLCE, DOLCE-COM, and Distributed Description Logic

Looking at the intended meaning of concepts in OR and ON, it is noticeable that both the concepts in OR and in ON relate to propositional attitudes, that is, beliefs, desires, and intentions. In OR, we classify the content of communicated requirements statements according to the speech act, and thereby also according to how the content relates to propositional attitudes. In ON, obligations/prohibitions act as external sources of motivation for norm-governed software agents, so that they appear to give rise to new intentions in agents. To map OR to ON by way of relationships of their respective concepts to propositional attitudes, we require an ontology in which propositional attitudes are explicit.

We adopt DOLCE [44] extended with Ferrario and Oltramari’s computational ontology of mind (COM) [23], and call the resulting ontology DOLCE-COM in this paper. COM is an add-on to DOLCE, in which propositional attitudes are defined within DOLCE and considered as kinds of DOLCE *mental attitude*, which in turn is a kind of *mental state*. A mental state therein is a kind of *state*, which itself is a *perdurant*. The advantage of choosing DOLCE is that we can relate our intended meaning to concepts in a foundational ontology, which allows future work in mapping other concepts in software engineering to those of OR and/or ON in the aim of further automation to reuse DOLCE and its extensions. DOLCE is chosen instead of other foundational ontologies because it rests on intuitively attractive ontological choices for the present discussion.

Being a descriptive ontology, categories in DOLCE are defined to reflect notions that depend on human perception, cultural imprints, and social conventions. Since requirements engineering is oriented towards the resolution of practical problems, in which decisions are informed by stakeholders with perceptual bias, individual cultural and other background, and specific social conventions, an ontology that does not recognize such bias could hardly be an appropriate choice. Such an ontology would be removed from

**Table 5.** Comparison of main foundational ontologies and their ontological choices (from [12]).

	BFO [58]	DOLCE [44]	OCHRE [54]	OpenCyc [16]	SUMO [29]
Descriptive	no	yes	no	yes	yes
Multiplicative	no	yes	unclear	unclear	yes
Possibilism	no	yes	yes	unclear	unclear
Perdurantism	yes	yes	no	unclear	yes

the reality of the problem that we intend to resolve. As its authors point out [44], DOLCE has a clear cognitive bias, for its ontological categories are intended to capture ontological categories that underlie natural language and human commonsense. This fits well with the approach adopted in defining our core ontology for requirements, as we define the concepts of our ontology by drawing from types of speech acts, which themselves are intended to reflect the communication of content expressed in natural language. Note that the requirements engineering effort involves the representation of information that is communicated by the stakeholders, and is therefore a process in which already formed conceptualizations are made explicit. Categories in DOLCE are defined to be descriptive notions that assist in rendering explicit already formed conceptualizations. In this respect, by being descriptive, DOLCE appears to be relevant to the RE process which itself is descriptive, that is, aims to capture what is perceived and communicated. The choice of DOLCE is grounded in essentially three arguments: (i) DOLCE is descriptive, and the modeling effort in RE appears to be descriptive as well; (ii) DOLCE takes a possibilistic view, which seems closer than the actualistic perspective on how reasoning is performed during requirements engineering (indeed, requirements engineering involves to a considerable extent the reasoning about what may and will be the case in the future, so that the ontology ought to be possibilistic); and (iii) the notion of quality, quality spaces, and quality values which seem to fit the intuition well and allows us to distinguish functional from nonfunctional requirements.

Foundational ontologies mentioned in Table 5 have been defined with different purpose in mind and therefore subsumes different ontological choices. The closest with regards to the needs presented herein are DOLCE and SUMO. DOLCE is preferred herein over SUMO as the former commits more clearly to the ontological choices deemed appropriate with regards to the problem at hand, that is, the definition of a core ontology for RE. SUMO (Suggested Merged Upper Ontology) aims to combine categories and relations coming from different top-level ontologies in order to improve interoperability, communication, and search in the Semantic Web field. Given that SUMO merges different upper level ontologies, it is not influenced by an overall theoretical approach, but adopts general categories from various sources. It is neither clearly multiplicative nor clearly reductionist, and the same dilemma applies to its position with regards to the choice of either possibilism or actualism. It is classified above as descriptive, since it includes the commonsense distinction between objects and processes.

We proceed in four steps to bridge OR and ON. First (Section 2.2), we bridge OR and DOLCE-COM by way of a helper ontology, in which speech acts are related to propositional attitudes. This will allow us to state, for example, that instances of DOMAIN-ASSUMPTION (a concept in OR) will involve instances of BELIEF (a concept in DOLCE-COM). Second (Section 2.3), we bridge ON and DOLCE-COM, which will allow us, e.g., to state that instances of OBLIGATION (in ON) will involve instances of INTEN-

TION (in DOLCE-COM). Because instances of OR relate to instances of propositional attitudes of the stakeholders, while instances of ON relate to propositional attitudes of the software agents, our third step (Section 2.4) consists of defining how instances of propositional attitudes of stakeholders map to instances of propositional attitudes of software agents. The fourth step (Section 2.5) combines the results of the first three steps to indicate how concepts in OR relate to those in ON.

Each of the ontologies is specified in OWL DL [53].<sup>5</sup> To relate concepts in separate ontologies, we use the distributed description logic (DDL) framework from Borgida and Serafini [4]. We briefly recall some key definitions and properties of DDL.

*DDL syntax.* Let  $I$  a set of indexes of ontologies that we intend to bridge; in our discussions below,  $I = \{DOLCE - COM, OR, ON, SA, AP\}$ . A *Distributed Description Logics* is a collection  $\{\mathcal{DL}_i\}_{i \in I}$  of Description Logics. Each ontology involved in bridgings is formalized by a T-box  $\mathcal{T}_i$  of  $\mathcal{DL}_i$ , so that all involved ontologies together correspond to a collection of T-boxes  $\mathcal{T} = \{\mathcal{T}_i\}_{i \in I}$ . A prefix is added to descriptions in each ontology in order to distinguish descriptions from different ontologies: e.g.,  $i : X$  denotes concept  $X$  from ontology  $i$ ,  $i : Y \sqsubseteq Z$  denotes  $Y \sqsubseteq Z$  specified in ontology  $i$ . Semantic relations between pairs of ontologies are specified in DDL using bridge rules. A *bridge rule* from ontology  $i$  to ontology  $j$  is an expression of one of the following two forms:

- *Into-bridge rule:*  $i : X \xrightarrow{\sqsubseteq} j : Y$ ;
- *Onto-bridge rule:*  $i : X \xrightarrow{\supseteq} j : Y$ .

where  $X$  and  $Y$  are, respectively, concepts of ontologies  $\mathcal{T}_i$  and  $\mathcal{T}_j$ . Conjunction of  $i : X \xrightarrow{\sqsubseteq} j : Y$  and  $i : X \xrightarrow{\supseteq} j : Y$  is abbreviated by  $i : X \xrightarrow{\equiv} j : Y$ . Intuitively,  $i : X \xrightarrow{\sqsubseteq} j : Y$  (resp.  $i : X \xrightarrow{\supseteq} j : Y$ ) indicates that, from the point of view of  $j$ ,  $X$  in  $i$  is more specific (resp. more general) than  $Y$  in  $j$ .

A *distributed T-box*  $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B} \rangle$  consists of a collection of T-boxes  $\mathcal{T} = \{\mathcal{T}_i\}_{i \in I}$  and a collection of bridge rules  $\mathfrak{B} = \{\mathfrak{B}_{ij}\}_{i \neq j \in I}$  between these ontologies.

*DDL semantics.* A *domain relation*  $r_{ij}$  represents a possible way of mapping the elements of  $\Delta^{\mathcal{T}_i}$  into the domain  $\Delta^{\mathcal{T}_j}$ :  $r_{ij} \subseteq \Delta^{\mathcal{T}_i} \times \Delta^{\mathcal{T}_j}$  such that  $r_{ij}$  denotes  $\{d' \in \Delta^{\mathcal{T}_j} \mid \langle d, d' \rangle \in r_{ij}\}$ ; for any subset  $D$  of  $\Delta^{\mathcal{T}_i}$ ,  $r_{ij}(D)$  denotes  $\bigcup_{d \in D} r_{ij}(d)$ ; for any  $R \subseteq \Delta^{\mathcal{T}_i} \times \Delta^{\mathcal{T}_j}$ ,  $r_{ij}(R)$  denotes  $\bigcup_{\langle d, d' \rangle \in R} r_{ij}(d) \times r_{ij}(d')$ .

A *distributed interpretation*  $\mathfrak{I} = \langle \{\mathcal{I}_i\}_{i \in I}, \{r_{ij}\}_{i \neq j \in I} \rangle$  of a distributed T-box  $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B} \rangle$  consists of a collection of local interpretations  $\mathcal{I}_i$  on interpretation domains  $\Delta^{\mathcal{T}_i}$ , one for each  $\mathcal{T}_i$ , and a family of domain relations  $r_{ij}$  between the local domains.  $\mathfrak{I}$  is said to satisfy  $\mathfrak{T} = \langle \mathcal{T}, \mathfrak{B} \rangle$ , denoted  $\mathfrak{I} \models \mathfrak{T}$  if all T-boxes in  $\mathcal{T}$  are satisfied and all bridge rules in  $\mathfrak{B}$  are satisfied:

<sup>5</sup>We share the usual reasons for choosing OWL DL: OWL has become the standard language for formalizing sharable knowledge and OWL DL is the most expressive dialect of OWL that remains decidable.

**Table 6.** Helper ontology of speech acts (denoted: SA).

108:	ASSERTIVE-SPEECH-ACT	$\sqsubseteq$	$\exists$ convey.BELIEF
109:	DIRECTIVE-SPEECH-ACT	$\sqsubseteq$	$\exists$ convey.DESIRE
110:	COMMISSIVE-SPEECH-ACT	$\sqsubseteq$	$\exists$ convey.INTENTION
111:	EXPRESSIVE-SPEECH-ACT	$\sqsubseteq$	PSYCHOLOGICAL-ATTITUDE
112:	DECLARATIVE-SPEECH-ACT	$\sqsubseteq$	$\exists$ convey.BELIEF
113:	REP-DECL-SPEECH-ACT	$\sqsubseteq$	$\exists$ convey.BELIEF
114:	refer-to	$\equiv$	referred-to-by <sup>-1</sup>
115:	referred-to-by	$\equiv$	refer-to <sup>-1</sup>
116:	T	$\sqsubseteq$	$\forall$ refer-to.QUALITY-VALUE $\sqcup \forall$ refer-to.WELL-DEFINED-QUALITY $\sqcup \forall$ refer-to.ILL-DEFINED-QUALITY
117:	T	$\sqsubseteq$	$\forall$ referred-to-by.DESIRE
118:	$\perp$	$\equiv$	WELL-DEFINED-QUALITY $\sqcap$ ILL-DEFINED-QUALITY
119:	convey	$\equiv$	conveyed-by <sup>-1</sup>
120:	conveyed-by	$\equiv$	convey <sup>-1</sup>
121:	T	$\sqsubseteq$	$\forall$ convey.BELIEF $\sqcup \forall$ convey.DESIRE $\sqcup \forall$ convey.INTENTION $\sqcup \forall$ convey.PSYCHOLOGICAL-ATTITUDE
122:	T	$\sqsubseteq$	$\forall$ conveyed-by.ASSERTIVE-SPEECH-ACT $\sqcup \forall$ conveyed-by.DIRECTIVE-SPEECH-ACT $\sqcup \forall$ conveyed-by.COMMISSIVE-SPEECH-ACT $\sqcup \forall$ conveyed-by.EXPRESSIVE-SPEECH-ACT $\sqcup \forall$ conveyed-by.DECLARATIVE-SPEECH-ACT $\sqcup \forall$ conveyed-by.REP-DECL-SPEECH-ACT
123:	communicate	$\equiv$	communicated-by <sup>-1</sup>
124:	communicated-by	$\equiv$	communicate <sup>-1</sup>
125:	T	$\sqsubseteq$	$\forall$ communicated-by.STAKEHOLDER
126:	T	$\sqsubseteq$	$\forall$ communicate.ASSERTIVE-SPEECH-ACT $\sqcup \forall$ communicate.DIRECTIVE-SPEECH-ACT $\sqcup \forall$ communicate.COMMISSIVE-SPEECH-ACT $\sqcup \forall$ communicate.EXPRESSIVE-SPEECH-ACT $\sqcup \forall$ communicate.DECLARATIVE-SPEECH-ACT $\sqcup \forall$ communicate.REP-DECL-SPEECH-ACT

$$\mathcal{J} \models T_i \text{ iff } \mathcal{I}_i \models X \sqsubseteq Y \text{ for all } A \sqsubseteq B \in T_i$$

$$\mathcal{J} \models i : X \xrightarrow{\sqsubseteq} j : Y, \text{ iff } r_{ij}(X^{\mathcal{I}_i}) \subseteq Y^{\mathcal{I}_j}$$

$$\mathcal{J} \models i : X \xrightarrow{\supseteq} j : Y, \text{ iff } r_{ij}(X^{\mathcal{I}_i}) \supseteq Y^{\mathcal{I}_j}$$

## 2.2. Step 1: Bridging OR and DOLCE-COM

Intended meaning for concepts in OR (see, Table 1) indicates that these concepts are defined in terms of speech acts. Once we know the speech acts in which some requirements statement is made, we can determine the propositional attitude that the stakeholder conveys by way of the speech act. To the best of our knowledge, an extension for speech acts on pair with COM in terms of rigor and detail is not available. DOLCE-COM features

BELIEF, DESIRE, and INTENTION; Ferrario and Oltramari provide an axiomatization thereof [23]. It is enough in this paper – without investigating an extension for DOLCE to cover speech acts – to know only the relationships between concepts in OR and propositional attitudes in DOLCE-COM for the simple reason that speech acts do not appear in mappings from ON to DOLCE-COM, as becomes apparent throughout the remainder of the paper. We cannot bridge directly OR and DOLCE-COM, since neither of these two ontologies is explicit about speech acts. Consequently, we introduce a simple “helper” ontology – denoted SA – whose sole purpose is to relate speech acts and propositional attitudes, and this according to Searle’s original taxonomy and informal definitions of speech acts [55,56], which we recalled earlier (see, Section 1.1).

The SA helper ontology is specified in Table 6. Axioms first relate kinds of speech acts with propositional attitudes that these speech acts convey. Propositional attitudes given there are defined locally in SA. Relationships are given to relate (i) speech acts with propositional attitudes they convey; (ii) relate desires and quality values and ill- or well-defined qualities, as required by the definitions of QUALITY-CONSTRAINT and SOFTGOAL in OR (see, Table 2); (iii) stakeholders with the communicated speech acts.

Given DOLCE-COM, OR, and SA, we first need to bridge DOLCE-COM and SA, then SA and OR. The meaning intended for the propositional attitudes in SA is the same as that of propositional attitudes in DOLCE-COM, which leads to the bridging rules 127–129 in Table 7(a). In DOLCE (as well as in DOLCE-COM), there are three kinds of QUALITY: TEMPORAL-QUALITY, PHYSICAL-QUALITY, and ABSTRACT-QUALITY. Given that any particular quality is either of these, the following axioms can be written:

$$\begin{aligned} \text{QUALITY} &\equiv \text{TEMPORAL-QUALITY} \sqcup \text{PHYSICAL-QUALITY} \sqcup \text{ABSTRACT-QUALITY} \\ \perp &\equiv \text{TEMPORAL-QUALITY} \sqcap \text{PHYSICAL-QUALITY} \sqcap \text{ABSTRACT-QUALITY} \end{aligned}$$

According to Table 2, an ill-defined quality is distinguished from a well-defined-quality in that the former involves a quality space having a subjective and/or ill-defined structure. While DOLCE remains neutral on the properties of temporal or physical qualities, we can admit for most practical purposes that these are neither subjective nor ill-defined. In most cases, the bridging rules 130–131 will therefore apply. A temporal or physical quality is thus necessarily a well-defined quality, whereas only an abstract quality can be either ill-defined or well-defined. QUALITY-VALUE delimits part of the quality space for the ill- or well-defined quality, so that it will for all practical purposes relate to ABSTRACT in DOLCE (and therefore DOLCE-COM) – thus, a quality value may be a set, a region, or some other kind of abstract in DOLCE.

Bridging rules in Table 7(b) result from the definitions in Table 2. Assertive, declarative, or representative declarative speech act communicated by a stakeholder and by which a belief is conveyed in requirements engineering gives an instance of a domain assumption (bridging rules 133–135). Directive speech act conveying a desire, and that refers to a well-defined quality and restricts that quality to some value gives us a quality constraint (136). A second case for a directive speech act arises when it refers to no well-defined quality nor quality value, whereby a requirements engineer interprets it as a goal (137). A directive speech act communicated by a stakeholder, conveys a desire, and if it constrains an ill-defined quality to some quality value, then it is a softgoal (138). A commissive speech act communicated by a stakeholder conveys an intention, thus appearing in requirements as a plan (139). Finally, an expressive speech act conveys a psychological attitude, which we then understand as an instance of ATTITUDE in OR (140).

**Table 7.** Bridging from DOLCE-COM to SA to OR.

(a) Bridging rules for DOLCE-COM and SA.		
127:	$DOLCE-COM:BELIEF$	$\equiv \rightarrow SA:BELIEF$
128:	$DOLCE-COM:DESIRE$	$\equiv \rightarrow SA:DESIRE$
129:	$DOLCE-COM:INTENTION$	$\equiv \rightarrow SA:INTENTION$
130:	$DOLCE-COM:QUALITY$	$\equiv \rightarrow SA:WELL-DEFINED-QUALITY$
131:	$DOLCE-COM:ABSTRACT-QUALITY$	$\supset \rightarrow SA:ILL-DEFINED-QUALITY$
132:	$DOLCE-COM:ABSTRACT$	$\equiv \rightarrow SA:QUALITY-VALUE$
(b) Bridging rules for SA and OR.		
133:	$SA:(ASSERTIVE-SPEECH-ACT$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:DOMAIN-ASSUMPTION$
134:	$SA:(DECLARATIVE-SPEECH-ACT$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:DOMAIN-ASSUMPTION$
135:	$SA:(REP-DECL-SPEECH-ACT)$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:DOMAIN-ASSUMPTION$
136:	$SA:(DIRECTIVE-SPEECH-ACT$ $\sqcap \forall \text{refer-to.WELL-DEFINED-QUALITY}$ $\sqcap \forall \text{refer-to.QUALITY-VALUE}$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:QUALITY-CONSTRAINT$
137:	$SA:(DIRECTIVE-SPEECH-ACT$ $\sqcap \neg \exists \text{refer-to.WELL-DEFINED-QUALITY}$ $\sqcap \neg \exists \text{refer-to.QUALITY-VALUE}$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:GOAL$
138:	$SA:(DIRECTIVE-SPEECH-ACT$ $\sqcap \forall \text{refer-to.ILL-DEFINED-QUALITY}$ $\sqcap \forall \text{refer-to.QUALITY-VALUE}$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:SOFTGOAL$
139:	$SA:(COMMISSIVE-SPEECH-ACT$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:PLAN$
140:	$SA:(EXPRESSIVE-SPEECH-ACT$ $\sqcap \exists \text{communicated-by.STAKEHOLDER})$	$\equiv \rightarrow OR:ATTITUDE$

Together, the bridging rules in Tables 7(a) and 7(b) allow us to relate instances of stakeholders propositional attitudes to instances of concepts in our ontology of requirements. In other words, these bridging rules mirror the requirements engineering process, in which stakeholders communicate using speech acts, conveying therefore their beliefs, desires, intentions, and psychological attitudes about the future MAS (as indicated by bridging rules in Table 7(a) clearly indicate); the requirements engineer can then instantiate OR in order to capture the content of the communication – and thereby the beliefs, desires, intentions, and psychological attitudes – of stakeholders, and this depending on the speech act (as pointed out in bridging rules in Table 7(b)).

### 2.3. Step 2: Bridging ON and DOLCE-COM

When bridging propositional attitudes to concepts in OR, the direction of bridging rules indicates that we start off from propositional attitudes, then move to speech acts, and

**Table 8.** Helper ontology of propositional attitudes and norm-governed agents (denoted: AP).

141:	INTENTION	$\equiv$	$\exists$ intended-by.NORM-GOVERNED-AGENT
142:	intend	$\equiv$	intended-by <sup>-1</sup>
143:	intended-by	$\equiv$	intend <sup>-1</sup>
144:	T	$\sqsubseteq$	$\forall$ intended-by.NORM-GOVERNED-AGENT
145:	T	$\sqsubseteq$	$\forall$ intend.INTENTION
146:	BELIEF	$\equiv$	$\exists$ believed-by.NORM-GOVERNED-AGENT
147:	believe	$\equiv$	believed-by <sup>-1</sup>
148:	believed-by	$\equiv$	believe <sup>-1</sup>
149:	T	$\sqsubseteq$	$\forall$ believed-by.NORM-GOVERNED-AGENT
150:	T	$\sqsubseteq$	$\forall$ believe.BELIEF
151:	condition	$\equiv$	conditioned-by <sup>-1</sup>
152:	conditioned-by	$\equiv$	condition <sup>-1</sup>
153:	T	$\sqsubseteq$	$\forall$ conditioned-by.BELIEF
154:	T	$\sqsubseteq$	$\forall$ condition.INTENTION
155:	$\exists$ condition.INTENTION	$\sqsubseteq$	$\exists$ believed-by.NORM-GOVERNED-AGENT

only then to requirements. This direction reflects the requirements engineering process.<sup>6</sup> Norm engineering has a different direction: because norms govern behavior, a norm is a means for affecting propositional attitudes of agents. Consequently, bridging rules in this section move from ON to DOLCE-COM, whereby we understand the propositional attitudes here as being those of norm-governed agents, and not of the stakeholders.

Obligations and prohibitions involve commitment, and we have noted earlier (see, Section 1.2) that our norm-governed agent takes obligations and prohibitions as external sources of motivation. Recall, from Cohen and Levesque [13], that an agent adopts intention X if (i) it believes X is possible, (ii) it does not believe it will not bring about X, (iii) it believes it will bring about X, (iv) it does not intend all the side effects of bringing about X, and (v) it invests effort in trying to bring about X. In summary, intention involves choice and commitment. It would thus seem that we can bridge OBLIGATION in OR to INTENTION in DOLCE-COM. Obligations and prohibitions would thus map to intentions, but only if the agent adopts them. Acceptance will depend on the result of the agent’s reasoning over its beliefs, desires, and intentions, and the obligation or prohibition that it is to evaluate. If the agent decides to adopt the obligation or prohibition, it will adjust its intentions in order to bring about the obliged or prohibited conditions. There is, however a nuance here: the propositional attitudes we speak of here are those of norm-governed agents. Notice that DOLCE-COM carries no notion of agent, while ON incorporates neither the agent nor the propositional attitudes. Consequently, a “helper” ontology – denoted AP – is needed here in order to relate norm-governed agents to propositional attitudes they can hold.

The helper ontology carries propositional attitudes and agents, and relates them in a usual way, in that intentions and beliefs are held by (resp. intended by and believed by) agents. Also, we allow beliefs to condition intentions, which is not controversial since agents consider beliefs in the process of choosing how to act.

<sup>6</sup>Obviously, some stakeholders may learn from requirements, so that requirements give rise to new propositional attitudes, but in the end, we are interested in what the stakeholders communicate – i.e., not what they may be thinking, but what they are expressing.

Permissions do not motivate behavior, hence cannot be related directly to desires. Recall from Section 1.2 that a permission allows condition  $c$  but disallows nothing (whereas an obligation allows  $c$  and disallows its opposite). Permissions do not commit, hence cannot relate directly to intentions. There are three cases we need to consider when discussing what permissions can serve for in a norm-governed MAS (or, in other words, how permissions can affect the propositional attitudes held by norm-governed agents):

1. Permissions can state the obvious: a permission to bring about  $c$  is subsumed in an obligation to bring about  $c$ , and inversely, a permission to  $\neg c$  is subsumed in a prohibition to bring about  $c$  (equivalently, obligation to  $\neg c$ ). Hence, if an obligation to  $c$  is explicit in a norm-governed MAS, giving a permission to  $c$  in addition amounts to repeat the obvious.
2. Permissions can remove constraints to act: say that there is no obligation to bring about  $c$ , but there is an obligation to bring about  $c'$ , whereby the agent committing to obligation on  $c'$  cannot meet this obligation without previously bringing about  $c$  ( $c$  may hold at some point in the process of bringing about  $c'$ ). Without committing agents, a permission on  $c$  will make it explicit that action can be taken to bring about  $c$ .
3. Permissions handle exceptions to obligations and prohibitions: if the obligation to bring about some condition  $c$  does not apply under some specific circumstances, then a permission will state that, under such circumstances,  $\neg c$  is allowed. Agents' failure to meet obligations or prohibitions are kinds of exceptions, and thus can be dealt with permissions: if the agent fails at meeting an obligation  $O_1$ , the permission will state that the agent is then allowed to attempt to meet some other obligation  $O_2$ .

Permissions used in the sense of 1 and 2 above map to beliefs of a norm-governed agent. The third case is quite different from the first two, leading a permission to relate to an intention – consider an example. Let  $O_1$  and  $O_2$  be two distinct obligations. The permission in the third case above may be written down as “if the failure conditions for  $O_1$  hold, try  $O_2$ ” and will be interpreted in the norm-governed MAS as indicating that any agent that adopts the obligation  $O_1$  and fails to meet that obligation must attempt to meet  $O_2$ . Notice that “if  $O_1$  fails, try  $O_2$ ” must be an intention, since trying  $O_2$  involves choice and commitment. The permission in this case amounts to an intention triggered by the failure to meet  $O_1$ . In practical terms, the obligation  $O_2$ , for which the permission “if if failure conditions for  $O_1$  hold, try  $O_2$ ” applies, will be adopted by the agent in such a way that the agent will take action only to meet  $O_2$  only if conditions indicating the failure of  $O_1$  verify. This way of relating permissions, beliefs, and intentions is practical for norm-governed MAS: it allows us to manage the behavior of agents when obligations or prohibitions are not met. Moreover, it is a mechanism for handling derogations to obligations and prohibitions. By bridging permissions to beliefs that trigger intentions, a permission can reflect precedence in obligations or prohibitions that arise from a category of legal principles, namely the rules of collision.<sup>7</sup> For instance, if  $O_1$  and  $O_2$

<sup>7</sup>Rules of collision include: *lex posterior derogat legi priori* (i.e., later norms prevail over earlier norms), *lex superior derogat legi inferiori* (i.e., norms superior by law prevail over inferior norms), *lex specialis derogat legi generali* (i.e., particular norms prevail over general norms), *lex superior prior derogat legi inferiori posteriori* (i.e., earlier superior norms prevail over later inferior norms), *lex superior generalis derogat legi inferiori speciali* (i.e., superior general norms prevail over inferior particular norms), and *lex prior specialis derogat legi*

**Table 9.** Bridging from ON to AP to DOLCE-COM.

(a) Bridging rules for ON and AP.		
156:	ON:OBLIGATION	$\xrightarrow{\sqsubseteq}$ AP:( $\exists$ intended-by.NORM-GOVERNED-AGENT)
157:	ON:PERMISSION	AP:( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcup \exists$ condition.INTENTION)

---

(b) Bridging rules for AP and DOLCE-COM.		
158:	AP:BELIEF	$\xrightarrow{\equiv}$ DOLCE-COM:BELIEF
159:	AP:DESIRE	DOLCE-COM:DESIRE
160:	AP:INTENTION	DOLCE-COM:INTENTION

are in conflict, and  $O_1$  is (legally) superior to  $O_2$ , then applying the legal principle *lex superior derogat legi inferiori* will lead to the conclusion that  $O_1$  strictly prevails over  $O_2$  – consequently, we can reflect the said legal principle in a permission by ensuring that  $O_2$  should be met only if failure conditions hold for  $O_1$ .

Table 9 gives the bridging rules between ON and AP, and AP and DOLCE-COM. The rules from ON to AP summarize the above discussion: obligations correspond to what the agent intends, while permissions to what the agent believes or what conditions agent’s intentions. To avoid bridging permissions to beliefs that condition intentions but which (the beliefs) are not held by an agent, the axiom 155 in Table 8 indicates that what conditions intentions is also believed by some agent.

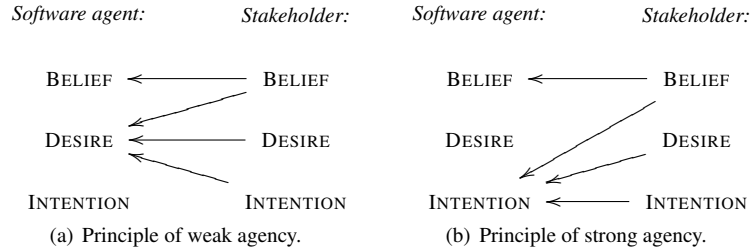
#### 2.4. Step 3: From Stakeholders to Norm-Governed Agents

The bridging rules defined up to this point establish how propositional attitudes of stakeholders relate to requirements (see, Section 2.2), and how norms relate to propositional attitudes of norm-governed agents (see, Section 2.3). Our aim now is to determine how the propositional attitudes of stakeholders relate to those of agents. Once this is known, we can suggest how to bridge OR and ON – as discussed in Section 2.5.

The most important observation at this point is that how propositional attitudes of stakeholders relate to those of the norm-governed agents is determined by the level of autonomy given to agents in the norm-governed MAS. Jensen and Meckling, in their seminal work in economics and finance on the theory of the firm [33], define the agency relationship “as a contract under which one or more persons – the principals – engage another person – the agent – to perform some service on their behalf, which involves delegating some decision making authority to the agent.” The stakeholders who provide the requirements clearly stand in this kind of relationship with the software agents that participate in the norm-governed MAS: the stakeholders are the principals, and the software agents are the Jensen and Meckling’s agents. Indeed, the MAS is intended to satisfy some requirements that the stakeholders would otherwise need to satisfy entirely through their own effort. Consider how the choice of relationships between the propositional attitudes of stakeholders and software agents determines the strength of the agency relationship (i.e., the level of autonomy of software agents):

---

*posteriori generali* (i.e., earlier particular norms prevail over later general norms). Rules of collision are one category of legal principles. Other categories, such as rules of interpretation used to establish the meaning of legal expressions, are not studied in terms of relationships over norms.



**Figure 1.** Mappings between propositional attitudes of stakeholders and software agents under two variants of the agency relationship.

- **Weak agency:** The agent may accept stakeholder’s beliefs as its own. If the beliefs are violated in the agent’s environment, the agent may adopt them as conditions to bring about and thus adopt them as desires. Stakeholder’s intentions may be adopted by the agent as desires. Whether the agent will act for the stakeholder will, in the case of weak agency, depend on what the agent concludes is the most appropriate way to act, given its own beliefs, desires, and intentions, and those from the stakeholder. There is, in other words, no commitment from the agent to act exactly according to the expectations of the stakeholder.
- **Strong agency:** Agents act on behalf of the stakeholders and adopt external sources of motivation – what motivates stakeholders motivates the software agents. A stakeholder’s beliefs, desires, and intentions captured by instances of OR correspond to motives according to which the software agent(s) act. It may seem appropriate then to assume that stakeholder’s beliefs map to agent’s beliefs, stakeholder’s desires to agent’s desires, and stakeholder’s intentions to agent’s intentions. This cannot, however be the case, because software agents reason on the basis of belief or knowledge (i.e., a belief or knowledge base) different from those upon which stakeholders’ form the beliefs, desires, and intentions that they communicate (for example, the consequence is that what is a domain assumption for a stakeholder is not a belief for an agent, but becomes a condition to ensure and thus an intention for the agent). Strong agency therefore means that stakeholder’s beliefs will either map to agent’s beliefs (if they verify in the agent’s environment), or as intentions, so that the agent acts to bring about the conditions believed by the stakeholder. As the agent commits to act for the stakeholder, the stakeholder’s desires and intentions will give rise to agent’s intentions.

Governance realized by norms, as in a norm-governed MAS, involves strong agency: if explicit commitment is absent (as in weak agency), the behavior of the software agents is not norm-governed. Strong agency fits the reasoning of our norm-governed practical reasoning agent outlined earlier (see, Section 1.2), which chooses behavior by evaluating the actions to take in light of its intentions and beliefs. Note in Figure 1(b) that stakeholder’s belief can map to agent’s belief or agent’s intention – the content of a stakeholder’s belief will be accepted by the agent as a new belief if it verifies in the agent’s environment; otherwise, the agent would adopt it as an intention in order to act and bring about conditions in which the stakeholder’s belief hold. Strong agency results in the bridging rules shown in Table 10. Bridging occurs between SA and AP as it is in these ontologies that propositional attitudes are associated, respectively, with stakehold-

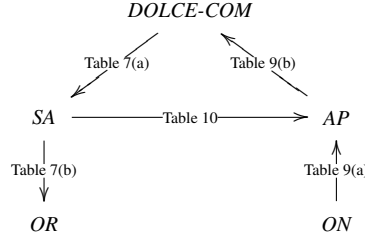
**Table 10.** Bridging rules from SA to AP resulting from the principle of strong agency. The direction of bridging rules – from SA to AP – reflects the position of stakeholders as principals, and the norm-governed agents as agents in the agency relationship.

161:	SA:(ASSERTIVE-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcup$ $\exists$ intended-by.NORM-GOVERNED-AGENT $\sqcap$ $\neg$ ( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcap$ $\exists$ intended-by.NORM-GOVERNED-AGENT))
162:	SA:(DECLARATIVE-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcup$ $\exists$ intended-by.NORM-GOVERNED-AGENT $\sqcap$ $\neg$ ( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcap$ $\exists$ intended-by.NORM-GOVERNED-AGENT))
163:	SA:(REP-DECL-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcup$ $\exists$ intended-by.NORM-GOVERNED-AGENT $\sqcap$ $\neg$ ( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcap$ $\exists$ intended-by.NORM-GOVERNED-AGENT))
164:	SA:(DIRECTIVE-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ intended-by.NORM-GOVERNED-AGENT)
165:	SA:(COMMISSIVE-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ intended-by.NORM-GOVERNED-AGENT)
166:	SA:(EXPRESSIVE-SPEECH-ACT $\sqcap$ $\exists$ communicated-by.STAKEHOLDER)	$\xrightarrow{\sqsubseteq}$	AP:( $\exists$ believed-by.NORM-GOVERNED-AGENT $\sqcup$ $\exists$ condition.INTENTION)

ers and with norm-governed agents. The direction of bridging rules from SA to AP reflects the position of stakeholders as principals, and the norm-governed agents as Jensen and Meckling’s agents. Whether the content of an assertive, declarative, or representative declarative speech act corresponds to beliefs of intentions of the agents depends on whether the agent establishes that the content in question verifies in the environment: if it does, then the agent adopts it as a belief, if it does not, then the agent ought to act (i.e., intend) to bring about the conditions conveyed by the speech act. The same rationale applies for expressive speech acts. For the rest, bridging rules reflect the strong agency relationship: desires and intentions of the stakeholders should lead agents to act.

#### 2.5. Step 4: Bridging OR and ON

Solid arrows in Figure 2 summarize the collections of bridging rules established up to this point. The rationale for these bridging rules was explained. The purpose of the specification of requirements is to make explicit in a precise manner the beliefs, desires, and intentions – broadly speaking, requirements – of the stakeholders, so that the MAS can be engineered to satisfy these requirements. The purpose of a specification of norms is to ensure that the norm-governed agents behave only in ways that lead to the satisfaction of the requirements. Consequently, in Section 2.2, we established how requirements reflect stakeholders’ beliefs, desires, and intentions. In Section 2.3, we showed how obligations, prohibitions, and permissions map to a norm-governed agent’s propositional attitudes –



**Figure 2.** Informal overview of the bridging rules. Solid arrows depict collections of bridging rules between the given ontologies. Directions of the arrows reflect the direction of the bridging rules. Arrow labels point to tables in the text where the corresponding bridging rules are defined.

in other words, we explained how norms influence the beliefs, desires, and intentions held by a norm-governed agent. In Section 2.4, we argued for how propositional attitudes held by stakeholders ought to be mapped to those held by norm-governed agents in order for norm-governed agents to act on behalf of the stakeholders.

Using bridging rules from SA to OR, and SA to AP that we can define bridging rules from OR to AP. For example, assertive speech act communicated by a stakeholder (as conceptualized in SA) corresponds to a domain assumption (as conceptualized in OR), as indicated by the bridging rule below (same as 133):

$$\begin{array}{l} SA:(\text{ASSERTIVE-SPEECH-ACT} \sqcap \exists \text{communicated-by.STAKEHOLDER}) \xrightarrow{\equiv} \\ OR:\text{DOMAIN-ASSUMPTION} \end{array}$$

According to the bridging rule below (same as 161) that follows the agency relationship principle, an assertive speech act communicated by a stakeholder (as in SA) is more specific than what is either believed or intended by a norm-governed agent:

$$\begin{array}{l} SA:(\text{ASSERTIVE-SPEECH-ACT} \sqcap \exists \text{communicated-by.STAKEHOLDER}) \xrightarrow{\sqsubseteq} \\ AP:(\exists \text{believed-by.NORM-GOVERNED-AGENT} \sqcup \exists \text{intended-by.NORM-GOVERNED-AGENT} \sqcap \\ \neg(\exists \text{believed-by.NORM-GOVERNED-AGENT} \sqcap \exists \text{intended-by.NORM-GOVERNED-AGENT})) \end{array}$$

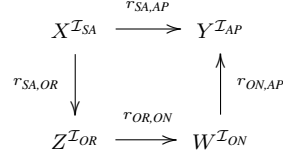
Let us rewrite the two bridging rules above as follows for the sake of readability:

1.  $SA:X \xrightarrow{\equiv} OR:Z$
2.  $SA:X \xrightarrow{\sqsubseteq} AP:Y$

Where we use the following abbreviations:

- $X \equiv \text{ASSERTIVE-SPEECH-ACT} \sqcap \exists \text{communicated-by.STAKEHOLDER};$
- $Y \equiv \exists \text{believed-by.NORM-GOVERNED-AGENT} \sqcup \exists \text{intended-by.NORM-GOVERNED-AGENT} \sqcap \neg(\exists \text{believed-by.NORM-GOVERNED-AGENT} \sqcap \exists \text{intended-by.NORM-GOVERNED-AGENT});$
- $Z \equiv \text{DOMAIN-ASSUMPTION}.$

Following the semantics of bridging rules and according to the first rewritten bridging rule,  $r_{SA,OR}(X^{\mathcal{I}_{SA}}) = Z^{\mathcal{I}_{OR}}$ , while  $r_{SA,AP}(X^{\mathcal{I}_{SA}}) \subseteq Y^{\mathcal{I}_{AP}}$  according to the second one. In other words, the first bridging rule indicates that instances of  $X$  in the domain of SA correspond to instances of  $Z$  in the domain of OR, and that every instance of  $Z$  in the domain of OR has a corresponding instance of  $X$  in the domain of SA. The second



**Figure 3.** Interpretations of concepts are shown according to their respective ontologies. Domain relations decorate arrows, whereby the arrows reflect the direction of bridging rules corresponding to the given domain relations. Arrows informally illustrate the direction of the “flow of information” between the ontologies.

bridging rule points out that instances of  $X$  in the domain of SA correspond to instances of  $Y$  in the domain of AP. Consider then the following bridging rule from OR to ON:

$$OR:Z \xrightarrow{\sqsubseteq} ON:W$$

The question in such a rule is what  $W$  is within ON, or, in other words, what instances in the domain of ON correspond to instances of DOMAIN-ASSUMPTION in the domain of OR. To find out, we use the above two bridge rules and add the following rules from ON to AP, established earlier (see, Section 2.3):

$$\begin{array}{l}
ON:PERMISSION \xrightarrow{\sqsubseteq} AP:(\exists \text{believed-by.NORM-GOVERNED-AGENT} \sqcup \exists \text{condition.INTENTION}) \\
ON:OBLIGATION \xrightarrow{\sqsubseteq} AP:(\exists \text{intended-by.NORM-GOVERNED-AGENT})
\end{array}$$

We know from Table 8 that:

- Axiom 1.4.6: BELIEF  $\equiv$   $\exists$ believed-by.NORM-GOVERNED-AGENT; and
- Axiom 1.5.5:  $\exists$ condition.INTENTION  $\sqsubseteq$   $\exists$ believed-by.NORM-GOVERNED-AGENT.

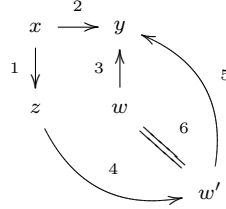
It follows that:

$$ON:(PERMISSION \sqcup OBLIGATION \sqcap \neg(PERMISSION \sqcap OBLIGATION)) \xrightarrow{\sqsubseteq} AP:Y$$

The meaning of the above bridge rule is that every instance of either PERMISSION or OBLIGATION in the domain of ON corresponds only to an instance of  $Y$  in the domain of AP. It follows that  $W$  is either a permission or an obligation. To better understand why, it is intuitive to think of bridging rules as describing “flows of information” from a domain to another domain [4] – i.e., a bridging rule from SA to OR describes a flow of information from SA to OR, so that, from the point of view of OR, OR is importing information from SA. The diagram in Figure 3 informally illustrates these flows of information.

The direction of arrows in Figure 3 reflects the direction of bridging rules. We can thus read the diagram in Figure 3 as follows: OR and AP import information from SA, ON imports information from OR, and AP imports information from ON. Hence, a belief (conveyed by an assertive speech act communicated by a stakeholder) in SA corresponds to a domain assumption in OR, and in AP to something either believed or intended by the norm-governed agent. Because that which is believed or intended by the norm-governed agent – in AP – is imported from ON, the bridging rule between OR and ON must ensure that what AP imports from ON corresponds to what AP imports from SA. This is why  $W$  is either a permission or an obligation, that is:

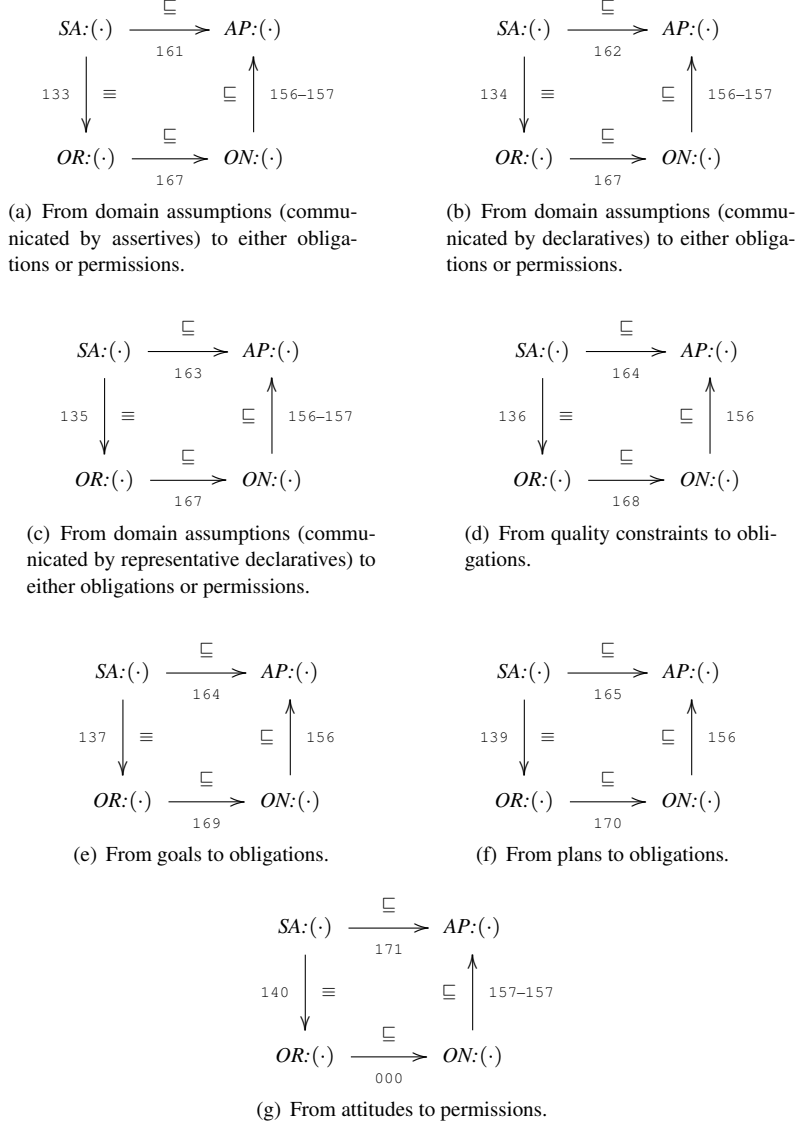
$$OR:DOMAIN-ASSUMPTION \xrightarrow{\sqsubseteq} ON:(PERMISSION \sqcup OBLIGATION \sqcap \neg(PERMISSION \sqcap OBLIGATION))$$



**Figure 4.** Arrows indicate correspondence between instances of concepts in order to illustrate the rationale behind the bridging rule  $OR:DOMAIN-ASSUMPTION \xrightarrow{\sqsubseteq} ON:(PERMISSION \sqcup OBLIGATION \sqcap \neg(PERMISSION \sqcap OBLIGATION))$ . For the meaning of numbers on arrows, see the text.  $x$  is an instance of  $X$  in the domain of SA;  $y$  is an instance of  $Y$  in the domain of AP;  $z$  is an instance of  $Z$  in the domain of OR;  $w$  is an instance of  $W$  in the domain of ON; and  $w'$  is an instance of  $W'$  in the domain of ON. The text explains why  $w'$  is equivalent to  $w$  and  $W$  is equivalent to  $W'$ .

The above is an into-bridge rule only (and not also an onto-bridge rule) because there may be permissions in a norm-governed that do not have corresponding domain assumptions – such permissions may be negotiated between agents or in general not originate from requirements. It should be noted that the above result is due to the application of the strong agency relationship principle. The bridging rule from SA to AP establishes that instances of the beliefs that stakeholders communicate bridge to instances of either beliefs or intentions, which must be reflected in the bridging from instances of OR to those of ON. In absence of bridging rules from SA to AP, there would be no criteria for choosing among potential bridging rules between OR and ON.

Another perspective may further clarify how we concluded that  $OR:DOMAIN-ASSUMPTION \xrightarrow{\sqsubseteq} ON:(PERMISSION \sqcup OBLIGATION \sqcap \neg(PERMISSION \sqcap OBLIGATION))$ . Consider the diagram in Figure 4, in which we consider individual instances. We build that diagram according to the following rationale (numbers on arrows in the diagram denote the reasoning step). If we have an instance of a belief that the stakeholder communicates – i.e., an instance  $x$  of  $X$  in the domain of SA – bridging rules from SA to OR indicate that that instance corresponds to an instance of a domain assumption – i.e., an instance  $z$  of  $Z$  in the domain of OR (arrow #1 in Figure 4). By the virtue of the bridging rules that reflect the strong agency relationship principle, that same instance of a belief that the stakeholder communicates – i.e.,  $x$  – corresponds to an instance of either a belief or an intention of a norm-governed agent – i.e., an instance  $y$  of  $Y$  in the domain of AP (arrow #2). Loosely speaking, this reflects the fact that the agent acts on behalf of the stakeholder. Now, based on the bridging rules from ON to AP, we know that to the instance  $y$  of  $Y$  in the domain of AP corresponds some instance  $w$  of the concept  $W$ , and in the domain of ON (arrow #3). The question we asked above is what concept  $W$  is in ON. Because we know that  $y$  is an instance of either a belief or an intention of a norm-governed agent,  $w$  is either a permission or an obligation and  $W$  is either PERMISSION or OBLIGATION. Note now that the instance, say  $w'$  – which is an instance of some concept  $W'$  in ON – to which  $z$  must correspond (arrow #4) should be such that  $w'$  corresponds to  $y$  in the domain of AP (arrow #5). Because  $x$  corresponds to  $y$ , and  $x$  corresponds to  $z$ ,  $w'$  is equivalent to  $w$  (arrow #6), and  $W$  is equivalent to  $W'$ . We therefore conclude that  $OR:DOMAIN-ASSUMPTION \xrightarrow{\sqsubseteq} ON:(PERMISSION \sqcup OBLIGATION \sqcap \neg(PERMISSION \sqcap OBLIGATION))$ .



**Figure 5.** Key bridging rules used to define the bridging rules from OR to ON. In each subfigure, the direction of arrows reflects the direction of bridging rules; each arrow references a bridging rule to be found in one of the Tables 7(b) (133–140), 9(a) (156–157), and 10 (161–166). Bridging rules shown between concepts in OR and ON result from the bridging rules between SA and OR, ON and AP, and SA and AP – the rationale bridging rules between OR and ON (summarized in Table 11). For readability, (·) replaces the relevant parts of each bridging rule – refer to the bridging rules referenced by numbers in the subfigures for the actual (·).

Applying the same straightforward reasoning – as that explained above to bridge domain assumptions to either permissions or obligations – leads us to bridge quality constraints, goals, and plans to obligations, while attitudes result in permissions. Hence, norm-governed agent, which adopts obligations that reflect the desires and intentions communicated by the stakeholders, acts in accordance to the strong agency relation-

**Table 11.** Bridging rules from OR to ON resulting from the bridging rules in referenced in Figure 5 and specifications of OR (see, Table 1), ON (see, Table 3), SA (see, Table 6), and AP (see, Table 8). The direction of bridging rules – from OR to ON – reflects the direction needed in converting requirements to norms in order the behavior of norm-governed agents in the MAS.

167:	<i>OR</i> :DOMAIN-ASSUMPTION	$\xrightarrow{\sqsubseteq}$	<i>ON</i> :(PERMISSION $\sqcup$ OBLIGATION $\sqcap \neg$ (PERMISSION $\sqcap$ OBLIGATION))
168:	<i>OR</i> :QUALITY-CONSTRAINT	$\xrightarrow{\sqsubseteq}$	<i>ON</i> :OBLIGATION
169:	<i>OR</i> :GOAL	$\xrightarrow{\sqsubseteq}$	<i>ON</i> :OBLIGATION
170:	<i>OR</i> :PLAN	$\xrightarrow{\sqsubseteq}$	<i>ON</i> :OBLIGATION
171:	<i>OR</i> :ATTITUDE	$\xrightarrow{\sqsubseteq}$	<i>ON</i> :PERMISSION

ship principle, that is, commits to act on the behalf of the stakeholder. Figure 5 shows the bridging rules that intervene in the definition of the bridging rules from OR to ON. For example, Figure 5(a) summarizes the bridging rules used in obtaining the bridging rule from domain assumptions – communicated by way of assertive speech acts – to either obligations or permissions. The rationale discussed earlier – in relation to Figures 3 and 4 – uses the bridging rules referenced in Figure 5(a) to arrive at the conclusion that domain assumptions correspond to either obligations or permissions. Applying this same rationale using bridging rules referenced in, e.g., Figure 5(d) leads us to conclude that quality constraints correspond to obligations. Figure 5 thus summarizes the bridging rules that are used to obtain the bridging rules from OR to ON, whereas Table 11 lists the obtained bridging rules from OR to ON. Note that SOFTGOAL in OR is not bridged, as it is – over the course of the requirements engineering process – approximated by quality constraints, so that bridging the approximating quality constraints to obligations ensures that the approximated softgoal will be satisfied.

### 3. Application: Automated Derivation of Norms from Requirements

The bridging rules in Table 11 indicate the correspondence from instances of concepts in the ontology of requirements to instances of concepts in the ontology of norms. These bridging rules provide a conceptual basis needed when defining algorithms to automate the specification of norms, given a specification of requirements for a norm-governed MAS. An all-encompassing algorithm would take in the specification of requirements, and produce a specification of norms. Such an algorithm would have the outline shown in Algorithm 1. There,  $c$  is an expression written using formal language constructs, and,  $r(c)$  is an instance of a concept in OR, whose content is  $c$  (e.g., a goal to bring about a state of affairs, in which the expression  $c$  holds). Depending on the concept instantiated to obtain  $r(c)$ , we follow the bridging rule from Table 11 relevant for that concept from OR, and create an instance of a concept from ON, to which we associate the content  $f(c)$ . The function  $f$  applied to the original content  $c$  may perform a transformation that is necessary for the norm-governed agent to be able to process the content – e.g., as when the domain ontology used by the agent is different than that used by the stakeholders. Thus, if  $r(c)$  is an instance of the GOAL concept from OR, we need to follow the bridging rule 169, and consequently, the algorithm (as shown in lines 10–11) creates an obligation with content  $f(c)$ . The agent, which subsequently adopts the newly created obligation

**input** : A consistent set, denoted  $R$ , of formally specified instances of concepts from the ontology of requirements (OR).

**output**: A set, denoted  $N$ , of formally specified instances of concepts from the ontology of norms (ON).

```

1 foreach requirement  $r(c)$  in  $R$  do
2   if  $r(c)$  is an instance of DOMAIN-ASSUMPTION then
3     if  $c$  does not verify in the environment of the agent then
4       Create an instance  $o(c')$  of OBLIGATION where  $c' \leftarrow f(c)$ , and insert  $o(c')$  in
5        $N$ ;
6     else if  $c$  verifies in the environment of the agent then
7       Create an instance  $p(c')$  of PERMISSION where  $c' \leftarrow f(c)$ , and insert  $p(c')$  in  $N$ ;
8     end
9   else if  $r(c)$  is an instance of QUALITY-CONSTRAINT then
10    Create an instance  $o(c')$  of OBLIGATION where  $c' \leftarrow f(c)$ , and insert  $o(c')$  in  $N$ ;
11  else if  $r(c)$  is an instance of GOAL then
12    Create an instance  $o(c')$  of OBLIGATION where  $c' \leftarrow f(c)$ , and insert  $o(c')$  in  $N$ ;
13  else if  $r(c)$  is an instance of PLAN then
14    Create an instance  $o(c')$  of OBLIGATION where  $c' \leftarrow f(c)$ , and insert  $o(c')$  in  $N$ ;
15  else if  $r(c)$  is an instance of ATTITUDE then
16    Create an instance  $p(c')$  of PERMISSION where  $c' \leftarrow f(c)$ , and insert  $o(c')$  in  $N$ ;
17  end

```

**Algorithm 1:** Outline of an algorithm that produces a specification of norms from a specification of requirements according to the bridging rules in Table 11.  $c$  is an expression written using formal language constructs, and,  $r(c)$  is an instance of a concept in OR, whose content is  $c$  (e.g., a goal to bring about a state of affairs, in which the expression  $c$  holds). The function  $f$  applied to the original content  $c$  may perform a transformation that is necessary for the norm-governed agent to be able to process the content – e.g., as when the domain ontology used by the agent is different than that used by the stakeholders.

will act in order to bring about  $f(c)$ , so that, once  $f(c)$  verifies, the stakeholders' goal is satisfied.

An example may be helpful at this point. Consider the transaction in which a Letter of Credit (LoC) is issued by a banker agent for use by a customer agent to finance the acquisition of goods from a supplier agent. The customer makes a deposit with the bank, then receives an LoC. The banker informs the supplier that an LoC has been issued to the customer, so that the customer can provide the LoC to the supplier, who can subsequently transfer the goods to the customer. To obtain the funds, the supplier sends the LoC to the bank. To engineer a norm-governed MAS for this setting, the first step amounts to requirements engineering, when stakeholders communicate their requirements, which the engineer subsequently analyzes – by way of available techniques in the field of requirements engineering for modeling (e.g., [47,17,11]), refinement (e.g., [18]), verification (e.g., [24]), negotiation (e.g., [2]), conflict resolution [61], and so on – to produce a formal requirements specification. That specification gives us the consistent set – denoted  $R$  in the algorithm in Algorithm 1 – of instances of concepts from the ontology of requirements (i.e., OR).  $R$  will contain part of the requirements specification, as softgoals are not converted to norms, but instead the quality constraints that approximate the softgoals (as explained earlier – see, Section 1.1).  $R$  thus contains instances of domain assumptions, goals, quality constraints, and attitudes, each instance carrying some content, that is, describing some logical condition. Assume that any logical condition  $c$  – that acts as content of instances of OR – is written as a Horn clause. A general form of Horn clause

is  $a_0 \vee (\neg a_1) \vee \dots \vee (\neg a_n)$ , and we adopt here the standard notation:  $a_0 \leftarrow a_1, \dots, a_n$ ; whereby  $a_0$  is true if  $a_1, \dots, a_n$  are true. Consider for instance the following condition:

$$eL : eLog, pL : paLog (logs(eL, pL, d) \leftarrow isPaperLog(pL, d), isElectronicLog(eL, d))$$

where  $eL, pL, d$  are instances of concepts defined in an ontology for the banking domain; namely,  $eL$  is an electronic log of a deposit,  $pL$  is a paper log of a deposit, and  $d$  is an identifier of a deposit made at the bank. Say that the above is a condition obtained by formalizing the content of a directive speech act made by some stakeholder. Consequently, the above would be the content of an instance of GOAL from OR. When the algorithm in Algorithm 1 considers that particular goal, an instance of OBLIGATION from ON is generated. The content of that obligation may be the same Horn clause, if  $f(c) = c$ , that is, if no transformation is needed.

Given the various kinds of requirements, we can usefully subdivide and consider parts of such an algorithm as separate algorithms, each specialized to a particular kind of requirement, e.g., goal, plan, quality constraints, and so on. Thus, an algorithm that generates an obligation from a goal is based on the hypothesis summarized as the bridging rule 169 in Table 11, whereas the algorithm that produces an obligation from a quality constraint relies on the bridging rule 168 in Table 11.

While bridging rules in Table 11 are straightforward, defining algorithms that obey such rules in actual settings is a complex matter. One difficulty is the potentially absent fit between the language used in the specification of requirements, and the language specifying norms and that agents can process. Another is the potential lack of fit between the domain ontologies used to describe the environment of the agents in the requirements specification and the specification of norms. We have discussed these issues, and suggested an overall approach to the derivation of norms from requirements, that includes algorithms that obey the above bridging rules. Both of the difficulties mentioned above are addressed by enforcing same ontologies in the requirements specification and the specification of obligations, prohibitions, and permissions. The resulting approach to the automated specification of norms from requirements – called Requirements-driven Contracting – is presented in detail elsewhere [36] (an introductory treatment is given in [40]). Therein, we introduce algorithms, which together correspond to that in Algorithm 1. The algorithms are adapted to a particular computational model of norm-governed MAS, along with specific languages for the specification of requirements and for the specification of norms. We do not pursue further discussion of the algorithms, and invite the interested reader to consult the cited texts [36,40].

#### 4. Related Work

The overall aim of the work reported in this paper is to understand how to facilitate the engineering of MAS. To this aim, the focus has been placed on understanding how the information acquired and analyzed during the requirements engineering phase of MAS development can be directly used in subsequent steps of the MAS engineering process, and in particular during the writing of norms that agents must obey when participating in a norm-governed MAS, that is a MAS in which obligations, prohibitions, and permissions govern the behavior of participating agents. Reusing requirements-level information during the engineering of norms is a two-part problem: (i) first, it is necessary to know how the various kinds of information relevant at the requirements level correspond to kinds

of information relevant during norm engineering; (ii) once this is known, it is possible to devise algorithms that create a norm specification from the information contained in a requirements specification. In this paper, the first (i) problem is formulated as one of ontology mapping, in which several ontologies – OR, ON, SA, AP, and DOLCE-COM – intervene and are combined according to a particular rationale – the core of which is the strong agency relationship principle (see, Section 2.4). We have argued above (see, Section 3) that the solutions to the second problem (ii) arise naturally after the bridging rules between an ontology of requirements and an ontology of norms are established, as these rules provide conceptual foundations for the definition of algorithms for the creation of norms from a specification of requirements. This paper focuses on the first problem (i), whereas we discussed in detail elsewhere the second (ii) problem [36,40].

The principal contribution of the present paper are the relationships – written as bridging rules – between our core ontology of requirements and a core ontology of norms. The value of these bridging rules has been discussed above: they serve as a conceptual foundation for the definition of algorithms for the writing of norm specifications from a specification of requirements.

To the best of our knowledge, efforts directly comparable to that of the present paper – that is, research on the bridging of requirements engineering and norm engineering by way of ontologies – are absent. Indirectly related work, which treats individually and either implicitly or explicitly some specific topic discussed in this paper, is available, and can be classified in the following categories: (a) frameworks for modeling and developing MAS, (b) frameworks for the engineering of virtual organizations (which are enabled by norm-governed MAS), (c) frameworks for specifying norms in MAS, and (d) efforts relating propositional attitudes of agents to concepts in implicit or explicit ontologies of norms for norm-governed MAS. We shall argue in the remainder of this section that in comparison to related work, our contribution is original because of the following:

- No available framework bridges requirements engineering and norm engineering.
- No available framework for the engineering of MAS or of virtual organizations can capture requirements as rich as those that can be captured by our core ontology of requirements.
- When an ontology of norms is available, it is not related to requirements.
- When an ontology of requirements is available or implicit, it is not related to an ontology of norms.
- Because of the above observations, no related effort has dealt with the problems of writing norms from requirements and automating such endeavor.

In the remainder of this section, we consider each of the categories of related work, and discuss how our contributions compare to these efforts.

#### 4.1. Frameworks for Modeling and Developing MAS

**OMNI.** The Organizational Model for Normative Institutions (OMNI) modeling framework for MAS [63] involves three dimensions: the normative dimension, specifying norms and rules to which members are expected to adhere; the organizational dimension, describing what roles are involved in the MAS, and how they interact; and the ontological dimension, defining relevant concepts of the application domain so as to facilitate the exchange of information among agents occupying the roles. Requirements

in OMNI take the form of “statutes” of the organization, that is, objectives, values, and the context (i.e., application domain) in which the organization operates. Objectives are associated to roles, along with rights, norms, and rules. OMNI provides three general ways for organizing the interaction between roles, namely the market (coordination by price), network (by trust), and hierarchy (by authority). Norms in OMNI cover obligations, permissions, and prohibitions, and are converted into rules, whose satisfaction can be verified. Violations lead to sanctions.

*Discussion.* The implicit ontology of requirements in OMNI is limited compared to OR presented here: (i) clear definitions of objectives and values are not available; (ii) preferences cannot be ported onto the governance level, and it is unclear how they are treated during MAS design; (iii) objectives are functional, so that it remains unknown how objectives relate to nonfunctional requirements. The ontology at the normative level is also implicit, although obligations, prohibitions, and permissions are all accounted for. Violation of norms is accompanied by sanctions. Overall, OMNI focuses on defining an organizational setting, in which agents can participate if they obey the norms. Because the requirements of the stakeholders, captured by the instantiation of our ontology of requirements, give rise to obligations, prohibitions, and permissions, and since OMNI features these notions at the normative level, the present work is compatible with OMNI: as stakeholders’ requirements are beyond the scope of OMNI, OMNI can take as input the norms that are obtained by the algorithms [36,40] that implement our bridging rules (defined in Section 1). This way, stakeholders’ requirements can be introduced among the norms for MAS organizations created through OMNI.

*Gaia.* The Gaia methodology [67] for the analysis and design of MAS starts with the identification of goals that the MAS is to satisfy. A model is then constructed of the environment in which the future MAS will operate. The environmental model describes resources available to the MAS and the actions that can be performed and that use the resources. The engineer then constructs a preliminary roles model, in which each role is defined as a collection of protocols and activities that the agent occupying the role can perform, permissions assigned through the role to the agent, and liveness and safety responsibilities. A model of interactions between roles is then built. The approach stops when the rules governing the overall operation of the MAS organization are defined. These indicate for instance, that some protocols can be executed only a given number of times, the sequence of protocols, and so on. Gaia does not commit to a particular formalism, leaving it to the engineer to choose a formalism relevant given the application domain at hand.

*Discussion.* Requirements are beyond the scope of Gaia. The form of governance in Gaia is left open, so that there is no predefined ontology of norms, whereas the notion of norm itself is absent. Instead, notions of permission and responsibility are used, and clear parallels with norms can be established: responsibilities state what agents must do, while permissions indicate limit on the resources that agents can exploit. Permissions and responsibilities define roles that agents can occupy. With respect to Gaia, the present work deals with a different level of abstraction, namely, the requirements of the stakeholders. Zambonelli and colleagues note that “Gaia could fit and be easily integrated with modern goal-oriented approaches to requirements engineering [11] whose abstractions closely match those of agent-oriented computing.” Our work on bridging requirements engineering and norm engineering fits precisely in this line of enquiry, and in this respect, the

norms that we obtain from requirements can be seen as Gaia's responsibilities (instances of the OBLIGATION and PROHIBITION concepts in ON) and permissions (instances of the PERMISSION concept in ON). As this paper illustrates, along with the accompanying work [36,40], integrating requirements with norm engineering – or, more generally, the engineering of MAS organizations – is not as easy as Zambonelli and colleagues expected.

**SODA.** The Societies in Open and Distributed Agent spaces [50] is a MAS design methodology, which involves analysis and design phases. In the former, role, resource, and interaction models are produced. The role model defines tasks that need to be performed, and tasks are allocated to roles, while roles are placed into groups. The resource model describes the MAS environment in terms of services (which exploit resources) that the environment can provide to the agents. The interaction model shows roles, groups, and resources interacting through protocols involving exchanges of information under interaction rules. For instance, a role is defined in terms of tasks, permissions to use resources, and an interaction protocol in which it takes part. At the design level, roles are mapped to agent classes, groups to societies of agents (defined by permissions, roles, and interaction rules), and resources to infrastructure classes (characterized by access modes, permissions, and allowed interaction protocols).

*Discussion.* SODA leaves requirements outside its scope. Normative aspects are missing, except for the permission concept. SODA can consequently be seen as dealing with levels of abstraction in MAS engineering lower than the normative one. While it may be interesting to study how the norms relate to SODA, more recent approaches, such as Gaia, and OMNI mentioned above are in this respect more promising.

**ISLANDER.** The ISLANDER approach [22] considers any agent-based organization as a setting in which message exchanges play a critical role, in the sense that they enable interaction. Several agents involved in interaction will participate in a scene, whereby a scene follows a precisely defined protocol. Primitive scenes are considered to be modules, and are composed into more elaborate scenes to guide complex interactions. In such a context, a role defines the scenes to which an agent can take part, and the protocols it should follow. ISLANDER requires a domain ontology, and involves ambient calculus for specification.

*Discussion.* Similarly to SODA, ISLANDER leaves requirements outside its scope. Normative concepts are missing. Focus is placed on the design of interaction protocols between agents, whereby constraints are defined on interactions to regulate agent behavior. Such constraints can be seen as obligations specific to agent interactions, defining roles for agents. The observation made for the SODA framework applies here: more recent approaches, such as Gaia and OMNI seem more promising for integration with the contributions we present herein.

**Tropos.** The Tropos methodology [11,24] spans the requirements, design, and implementation phases of MAS development. It features rich requirements models, grounded in the *i\** modeling framework [66]. Functional and nonfunctional requirements and domain assumptions are covered, along with roles and patterns of organized interaction [43]. Dependency models are used at the early requirements phase for preliminary structuring and documentation of natural language requirements. These models allow the engineer to study the impact that the MAS is likely to have within the environment in which

it will be deployed. Early requirements are formalized using a linear temporal first-order logic and model checking can be performed on specifications of restricted size [24]. Late requirements phase involves the definition of a MAS architecture in terms of roles and their interdependencies, whereby the architecture is chosen so as to best satisfy nonfunctional requirements. Detailed design involves the definition of data schemas, tasks, and interactions, using, e.g., the Unified Modeling Language. Implementation involves the definition of intelligent agents needed to populate the MAS.

*Discussion.* We have discussed at some length elsewhere [34,35,36,37,40] that the ontology of requirements that underlies the requirements models, such as those of Tropos, need to be enriched with notions of preference, along with a finer taxonomy of requirements, and corresponding goals. Tropos incorporates concepts that cover part of OR: namely, goal, softgoal, quality constraint, and domain assumption. Tropos can therefore be used as a requirements engineering framework when producing the requirements specification, although at present it does not cover the ATTITUDE concept in OR. Tropos does not consider that MAS are governed but instead assumes that appropriate agents are designed or otherwise found to occupy roles. Consequently, there are no normative concepts in Tropos. It is important to note here that, while Tropos covers requirements engineering, architectural design, and the design of individual agents, Tropos does not rely on explicit ontologies at all steps of software engineering that it covers, and it does not bridge these ontologies as we do here with OR and ON. Moving from one step to the other is left to the intuition of the software engineer, and some assistance is provided in the form of informal heuristics. Conceptual foundations for bridging the various levels are thus weak.

***Agent interaction governed by Business Protocols.*** Desai and colleagues' [19] approach to managing interactions of agents in a MAS involves the definition of business protocols. A business protocol is "an abstract, modular, and publishable specification of rules that govern a business interaction among two or more roles." [19] A protocol that satisfies some specific business goal is modular; the protocol is also abstract if it does not model proprietary business logic. Protocols can be combined to satisfy more abstract goals. Roles are defined within a protocol, whereby each role is defined in terms of specific interactions in which the agent occupying the role is involved when in the protocol. Interactions in a protocol involve the creation, satisfaction, and transformations of commitments. A commitment  $C(x, y, p)$  denotes that the agent  $x$  is obliged to agent  $y$  to bring about condition  $p$ . Desai and colleagues suggest that their approach complements methodologies for MAS development, such as Tropos and Gaia, by providing them with protocols that can be used as building blocks of organizational processes.

*Discussion.* In terms of the present paper, Desai and colleagues' commitment  $C(x, y, p)$  could be interpreted by saying that  $x$  is a software agent,  $y$  a stakeholder, and  $p$  the content of an instance of OBLIGATION from ON. This is quite limiting, however, since we are relating specific agents to specific stakeholders, so that we assume that individual agents are held accountable for the satisfaction of stakeholders' norms. Desai and colleagues' notion of commitment is considerably more useful once both  $x$  and  $y$  are software agents, as they indeed are intended to be in the cited work. When protocols are defined, it is assumed that the functional and nonfunctional requirements are known, so that requirements remain outside the scope of Desai and colleagues' approach. The implicit norm ontology they use involves obligations only. Just as OMNI and Gaia, Desai

and colleagues' approach is complementary to our work here: content of commitments can originate in the obligations that we obtain from requirements. Carrying over the attitudes of the stakeholders to commitments is, however, not obvious given the current state of the implied norm ontology in Desai and colleagues' approach [19].

**Norms in SMART.** López y López and colleagues [64,65] use the SMART agent framework [20] to provide a formal model of norms, so as to enable the specification of norms and facilitate their subsequent implementation. A norm is specified by instantiating a norm template. A norm template specifies a set of normative goals – i.e., conditions to bring about – addressed to a set of addressee agents by a set of beneficiaries – i.e., agents that benefit from the satisfaction of the normative goals. Sanctions and rewards are specified as well, to indicate the effects for the addressee agents of respectively, the violation or satisfaction of the norm. Preconditions for the norm are also given, along with the exceptions that identify states of affairs in which the normative goals should not be pursued by the addressee agents.

*Discussion.* López y López and colleagues do not study the relationships between norms and requirements. The components of a norm identified by López y López and colleagues provide an implicit concept of norm and thereby assume an ontology of norms. Compared to ON, we can say that their norm ontology includes obligations and prohibitions, and thereby partly encompasses ON. It is clear that normative goals correspond to the content of our instances of obligation or prohibition (recall that a prohibition can be reformulated as an obligation and *vice versa*). Our permissions are, however partly covered, since exceptions in López y López and colleagues' norm templates allow us to deal with permissions that aim at handling exceptions to norm, but not the permissions that are unrelated to obligations. Provided that the López y López and colleagues' norm template is adjusted to accommodate permissions, it can be related to our work in the same sense as OMNI and Gaia, playing the role of the framework in which the norms are specified, whereby our work provides means for obtaining the content of norms.

*Conclusions regarding frameworks for modeling and developing MAS.* The following conclusions can be drawn after relating the present work with that of the above frameworks for modeling and developing MAS:

- Frameworks for the modeling and specification of organizations in MAS based on behavior regulation by norms (above: OMNI, Gaia, Business protocol-based agent interaction) leave outside their scope the requirements engineering effort. Our work here clearly complements these frameworks in that it bridges the requirements specification to the specification of norms. Further investigation is needed to study the combined use of the results here (and algorithms presented elsewhere [36,40]) with OMNI and Gaia. If permissions can be incorporated in business protocols, the combined use of our contributions with Desai and colleagues' approach [19] should be investigated. If López y López and colleagues' approach [64,65] is extended with permissions, integration with our work should be explored. Other mentioned frameworks for MAS either do not incorporate norms, or include notions that appear close to norms. Hence, Gaia and OMNI seem at present more promising for combination with the present work.
- Disregarding the fact that Tropos is not focused on norm-governed MAS, Tropos remains the only currently available framework for MAS engineering that cov-

ers requirements engineering, architectural design, and the design of individual agents. Moving between these tasks is, however, based on informal heuristics and not grounded in bridged ontologies. Without competing with Tropos, we do make two important contributions that hopefully will be carried over to reinforce the research on Tropos. First, we show above and elsewhere [36,40] that it is relevant to relate two steps in MAS engineering on the basis of bridged ontologies, for it gives more solid links between the levels of abstraction traversed over the course of MAS engineering. Second, it explicit ontologies at each of the levels of abstraction (i.e., in Tropos: requirements, architecture, individual agent design) are necessary if we are to define the bridging rules and facilitate by further automation the MAS engineering effort.

#### 4.2. Frameworks for the Engineering of Virtual Organizations

A virtual organization (VO) [49] enables various heterogenous agents to coordinate resource sharing and problem solving activities. In their most generic formulation, VO's coordinate many software and human agents when these engage in sophisticated forms of interaction. Norm-governed MAS enable VO's. In this respect, frameworks for the engineering of VO's can be perceived as having a similar role to the frameworks for the modeling and development of norm-governed MAS.

**CONOISE-G.** CONOISE-G [48] is a framework incorporating agent-based models and techniques needed for the automated formation and maintenance of VO's. The framework covers the entire lifecycle of a VO, including formation, operation, perturbation, and dissolution phases. At formation, a requirements agent represents the user, acquires user's requirements, then initiates the formation of a VO by distributing a call for bids for the satisfaction of the specific requirements. Once the deadline for the proposals passes, the requirements agent selects a combination of agents best capable to fulfil the requirements. After formation, the agents act and interact in order to fulfil the requirements, whereby their actions are monitored to ensure expected quality of service levels. Perturbation occurs when better agents become available or because the current members do not perform as expected and agreed. Agents are replaced accordingly. The VO dissolves once the requirements have been fulfilled, and agents are no longer needed.

*Discussion.* CONOISE-G does not focus on the requirements level; it is concerned about the finding of optimal groups of agents to fulfil functional user requests. Nonfunctional considerations are catered through QoS constraints enforced by the monitoring agents. It is unclear whether the VO is governed or managed in some other way. Ontologies are only implicit, and while functional and nonfunctional constraints can be defined, the treatment of preferences seems to be missing. Normative concepts are not explicit. Transformation of requirements onto lower-level notions is also not discussed, seemingly indicating that requirements are already specified and that their format is directly appropriate for managing agents selected by the requirements agent.

**ADEPT.** The Advanced Decision Environment for Process Tasks [31] views a business process as a collection of autonomous agents that interact in order to reach mutually acceptable agreements that coordinate their interdependent subactivities. Agents manage services, that is, conceptual units of problem-solving activity in the business process. ADEPT simplifies the design of business processes by automating the allocation,

scheduling, and execution decisions that are specified at design time in non-agent-based systems. Each service is described with a unique ID, and the specification of inputs, outputs, guard, and body. Service level agreements specify contractual terms regulating interactions between agents.

*Discussion.* ADEPT starts off from already defined contracts – hence it assumes that obligations, prohibitions, and permissions are already defined. The relationships between the requirements that led to the provisions in the contract are not studied. Governance notions are limited, being restricted to obligations and sanctions. Ontologies for contracts or requirements are not explicit.

*Conclusions regarding frameworks for the engineering of VO's.* The above two approaches lack ontological bases for norms and requirements. Remarks voiced earlier on frameworks for the modeling and developing MAS can be repeated here, that is, our work may prove compatible with CONOISE-G and ADEPT. However, in order to establish such compatibility, the ontologies used in CONOISE-G and ADEPT ought to be made explicit. Ideally, not only would they be made explicit, but they would also be bridged to a foundational ontology such as DOLCE. At that point, both of these frameworks would become more interesting both to requirements engineers and MAS engineers, as both are interested in how the information acquired during requirements engineering is carried over to the level at which CONOISE-G and ADEPT apply.

#### 4.3. Frameworks for the Definition of Norms in MAS

**OCA.** The role-based model for Organized Collective Agency [52] involves the use of a deontic and action modal logic to capture the concept of acting in a role and relating it to obligations, permissions, and prohibitions. An organization in OCA consists of roles, deontic characterizations of roles (i.e., obligations, permissions, and prohibitions intrinsic to the role), transmissions of obligations (describing how roles are assigned to agents—this amounts to stating that whenever an agent has some specific obligation, it necessarily holds the role to which the obligation is assigned), and relationships between roles (sub-role, implication to ensure that if some role is held then another one is held as well, and incompatibility of roles so that they cannot be assigned simultaneously to the same agent). Interactions between agents are governed by contracts. A contract identifies the agents involved, the roles they occupy, and the obligations, permissions, and/or prohibitions of the roles.

*Discussion.* The ontology implicit in OCA includes all the concepts from ON. Additional concepts – primarily that of contract and of role – are needed to govern the interactions of the norm-governed agents. We do not consider these additional concepts herein, as we are interested only in how instances of norms – i.e., obligations, prohibitions, and permissions – need to be defined in order to reflect appropriately the requirements communicated by the stakeholders. ON carries only core concepts, whereas additional ones, such as those used in OCA, are certainly needed for the appropriate governance in MAS, but are defined over the core concepts: for instance, a contract involves roles, each role being defined by obligations, prohibitions, and permissions. OCA focuses on the definition of roles and their relationships, so that it can take as given the norms that we produce from the requirements, and by way of bridging rules.

**LGI.** “Law Governed Interaction is a mode of interaction that allows a heterogenous group of distributed agents to interact with each other, with confidence that an explicitly specified set of rules of engagement—called the law of the group—is complied with.” [46] The law of the group is an explicit and enforced set of rules that are enforced among the agents participating in the group. The law defines roles of agents, and their associated obligations. Sanctions apply when obligations are not held. Laws are written in Prolog, and the the coordination mechanism on which laws can be defined is implemented in the Moses toolkit [45].

*Discussion.* Again, as for much of mentioned related efforts, the higher levels of abstraction, namely those of requirements and the corresponding specification, are not considered. The implicit ontology of norms is limited to obligations. The advantage of LGI over OCA is that the former is more attractive in computational terms, while the latter is more expressive and specialized for norm-governed MAS.

*Conclusions regarding the frameworks for the definiton of norms in MAS.* The frameworks for the definition of norms focus on the dynamics of interactions between norm-governed agents and constrain these interactions by the definition of roles that the norm-governed agents adopt in order to participate in the MAS. Our work can be seen as dealing with obtaining norms from requirements, wheras these norms will define the roles, and thereby the contracts that the agents sign upon entering the MAS.

#### 4.4. *Efforts Relating Propositional Attitudes of Agents to Norms in Norm-Governed MAS*

We chose in Section 1.2 to consider agents with beliefs, desires, and intentions as propositional attitudes, and to consider norms as being imposed to agents from outside with the effect of changing agent’s intentions (in case of obligations and prohibitions) or beliefs (in case of permissions). The aim at present is to look at other ways of relating norms and propositional attitudes of norm-governed agents and discuss how these relate to the choice made in the present paper.

**BOID Architecture.** Broersen and colleagues [8] introduce obligations as an additional propositional attitude, and suggest an agent architecture in which beliefs, obligations, intentions, and desires coexist. Given beliefs and intentions, the agent’s problem lies in deciding which desires or obligations to follow. The reason for introducing obligations inside the set of propositional attitudes is that obligations can be violated due to the autonomy of the agent. This leads to several types of agents, which vary according to the relative priority given to obligations or other propositional attitudes. “In a realistic agent, beliefs override obligations, intentions or desires; in a single-minded or stable agent intentions override desires and obligations; in an open-minded or unstable agent desires and obligations override intentions; in selfish agents desires override obligations and in social agents, obligations override desires.

*Discussion.* If one chooses the BOID architecture, grounding it in DOLCE becomes more difficult as the concept of obligation should be found within the foundational ontology. Regardless of the controversy in equating the status of obligations to propositional attitudes, it is not clear why it is necessary to introduce obligations within propositional attitudes. If autonomy is sought, then let the agent decide whether to adopt the obligation

or not. If not, it will not participate in the norm-governed MAS, if it does, then it will act as a social agent, in which obligations govern behavior. By definition, acting within an organization limits the potential actions. Letting the agent cheat by adopting obligations in order to let it into the norm-governed MAS, then having that same agent behave against the obligations is rather uninteresting from the perspective of a stakeholder who spends resources to have the MAS satisfy some goal. Clearly, having these different types of agents corresponds to the reality of human organizations, but it still fails to justify placing obligations within propositional attitudes. Another option, which could enable selfish, open-minded, or other behavior is to have the agent conditionally adopt obligations, that is, transform them during adoption, e.g., by adding a precondition based on its beliefs or desires (e.g., act according to obligation  $O$  if it does not obstruct desire  $D$ ). Our understanding of the norm-governed agent is conservative: we assume that the agent has beliefs, desires, and intentions, and that it adopts obligations as new intentions. In this, we understand that adopting obligations involves choice and commitment, which, as Cohen and Levesque [13] convincingly argued, is what intentions involve as well.

**Normative beliefs and normative goals.** According to Conte and Castelfranchi [14,15], norms are aimed at controlling the behavior of agents. The control becomes possible when the addressee of a norm agent adopts it: a normative belief is created, which represents the belief about what ought to be brought about; a normative goal is generated from the belief to guide the behavior of the agent. Norms thus impact goal generation, in that they orient the agent's reasoning by indicating the normative goals to take into account. Also, norms impact goal selection, by providing criteria for comparing alternative goals – namely, norms establish (preference) orders over goals. Furthermore [10], norms can impact plan generation and selection, by excluding some actions, or by defining a preference order over plans. The agent is, in this perspective, able to deliberate on whether to obey the norm.

*Discussion.* Conte and Castelfranchi's agents are deliberative, that is, can decide whether to adopt norms, and if they adopt them, whether to act fully in compliance to norms. In the terminology of the present paper, their agents act according to a variant of the weak agency relationship principle (see, Section 2.4). The question here is: do our bridging rules need revision if the agents are deliberative, such as those of Conte and Castelfranchi? Once we adopt deliberative agents, what does change are the bridging rules from ON to AP: norms no longer bridge to intentions, but to desires and/or beliefs (depending on the precise choices in the reasoning mechanism of the norm-governed agent). However, what clearly cannot change is the stance that stakeholders call on norm-governed agents in order to have the requirement satisfied. In this respect, choosing a deliberative agent – instead of a more obedient one that we used in this paper – does not eliminate the basic principle that leads to how OR map to ON: regardless of the freedom in deliberation of the agents, the stakeholders still expect the agents to act on their behalf, and still expect to influence the agents in the way that will lead the latter to act in ways that satisfy requirements. Hence, if one chooses a more autonomous agent, the principal consequence is that there is more uncertainty that agents will indeed act according to norms. Even though bridging rules from SA to AP and from ON to AP will change to reflect the agent's ability to deliberate and not act blindly according to the requirements of the stakeholder, the adjustment will be such that the bridging rules from OR to ON will still apply (see, Table 11). For instance, if we have the deliberative agent, the con-

tent of a directive speech act communicated by the stakeholder will no longer bridge to what is intended by the agent, but to what is desired by the agent (so that the agent can choose whether to act upon that desire). This is undoubtedly weaker and introduces more uncertainty (all else being equal) for the stakeholder compared to the strong agency relationship. But this change must be cascaded onto bridging rules from ON to AP as well: if the stakeholder's desires can only be reflected in the agent's desires (and not intentions), and if the obligation is the only way to convey to the agent what ought to be done, then the obligation would no longer bridge to what is intended by the agent (since intentions cannot be influenced), but to what is desired by the agent (since this is now what can be influenced by norms). Consequently, our bridging rules between OR and ON will still remain standing. What would change as a consequence of increased autonomy – i.e., ability to deliberate on norms – is the strength of the agency relationship, but as explained, this would cascade changes to the bridging rules from ON to AP, and SA to AP, so that we will still have that stakeholders' goals bridge to obligations, quality constraints to obligations, attitudes to permissions, and so on. From another perspective, this simply means that we have taken in this paper an ideal case from the perspective of the stakeholders, in which strong agency guarantees that agents to some extent blindly act as the stakeholders command. Still, a different case – that of deliberative norm-governed agents – does not bring the established bridging rules from OR to ON down. Following this remark, it is important to note that Conte and Castelfranchi's work does not deal with the bridging of requirements and norms, and since their deliberative agent can be accommodated through the adaptation of bridging rules, our work is compatible with MAS architectures that rely on their principles for norm deliberation.

**Regulative and Constitutive Norms in Normative MAS.** Boella and van der Torre [3] introduce a formal framework for the construction of norm-governed MAS. A distinctive characteristic of that framework is the distinction between regulative and constitutive norms (following Searle [55]): regulative norms describe obligations, prohibitions, and permissions, while constitutive norms regulate the creation of institutional facts, such as, e.g., property. The constitutive norms enable economy in the specification of conditions for the applicability of regulative norms – regulative norms indeed require preconditions or triggers, and the specification of these can be facilitated by the classification of sets of states of affairs (or conditions) as constitutive norms. An obligation results in a goal attributed to an addressee agent, whereby the obligation also specifies the violation conditions, and the corresponding sanctions. Prohibitions are obligations concerning negated conditions, as usual. Permissions are defined as exceptions to obligations. The interaction of agents is conceptualized as a game.

*Discussion.* Boella and van der Torre's agents are deliberative, in a similar sense to those of Conte and Castelfranchi, so that the same arguments apply with regards to the weak vs. strong agency relationship. Obligations, prohibitions, and permissions form the implied ontology of regulative norms in Boella and van der Torre's approach, so that our work remains compatible with theirs: we provide means to obtain the content of obligations, prohibitions, and permissions from stakeholders' requirements. Our core ontology of norms covers regulative norms, but not constitutive ones. As Boella and van der Torre observe "assertions describing facts are constitutive norms", so that some domain assumptions would result in constitutive norms. Note that Boella and van der Torre define permissions exclusively as means to specify exceptions to obligations (or

prohibitions), so that their permissions are more specific than ours. Constitutive norms fall within our notion of permission. More precisely, when we use permissions to state facts that allow behavior independently of obligations or prohibitions, we can see these permissions as constitutive norms, especially if the permissions have broad application, as constitutive norms have by definition.

*Conclusions regarding efforts that relate propositional attitudes of agents to norms in norm-governed MAS.* Two conclusions should be highlighted. First, our choices in terms of relationships between norms and propositional attitudes is coherent with accepted and uncontroversial results, so that there is no barrier to combine our work with agent architectures in which agents can interpret standard kinds of norms – i.e., obligations, prohibitions, and permissions. Second, there is no enquiry similar to the present one – that is, studies of norms and propositional attitudes consider the problems of agent’s adoption of norms and subsequent behavior adaptation, so that sources of norms – in particular, the requirements of the stakeholders of the norm-governed MAS – are not discussed. The advantage of our approach with regards to the discussions of agent’s reasoning in the face of norms, is that our final bridging rules (those from OR to ON) are robust with regards to the level of autonomy (i.e., freedom in deliberation) of the agents, and that our core ontology of norms carries standard notions.

## 5. Conclusions and Future Work

Behavior of agents in norm-governed MAS is regulated by the definition of norms, that is, obligations, prohibitions, and permissions. The purpose of norms is to guide the behavior of agents towards the satisfaction of the requirements expressed by the stakeholders of the MAS. Norms therefore must be defined in accordance to the requirements. The premise of this paper is that a significant part of a specification of norms can be obtained from a specification of requirements, provided that the relationships between concepts involved in the specification of norms, and those involved in the specification of requirements are known. To establish these relationships, we have outlined a core ontology of requirements and a core ontology of norms (Section 1), used distributed description logics to define bridging rules between the concepts in the ontologies of requirements and of norms, and this by way of helper ontologies, themselves bridged to the DOLCE foundational ontology (Section 2). The result are bridging rules from the core ontology of requirements to the core ontology of norms, which – as we illustrated – subsequently serve as conceptual foundations for the definition of algorithms that take in a specification of requirements and produce a specification of norms (Section 3).

As we have argued in light of related work (Section 4), this research is novel in several respects. Salient features depend on the reader’s perspective:

- *Norm-governed MAS perspective:* While the literature on norm-governed agents and MAS studies extensively the problems of norm modeling, specification, consistency, change, and adoption, the sources of norms are not explored. In this respect, we complement the various works on norm specification by studying the requirements of the stakeholders of the MAS as an important source of norms. We thus provide a bridge between requirements engineering and norm engineering, which is clearly necessary for the successful development and use of norm-governed MAS.

- *Applied ontologies perspective:* From the perspective of research in ontologies for information systems, this paper brings no new theoretical ideas: several ontologies are specified using a standard formalism, and an available framework – that of distributed description logics – is then used to bridge these ontologies. This paper is in this respect a case study in the use of ontologies in the process of engineering MAS. Although only a case study, it does have a distinguishing trait: we use distributed description logics to bridge ontologies that underlie different steps – some would say different “levels of abstraction” – in MAS engineering. This is somewhat different from the problem of, say, bridging distinct ontologies of different university libraries or of different banks. The meaning behind requirements is rather different from that of norms, so not only is there significant syntactic mismatch here, but also a semantic one and a pragmatic one. This is why we sought the rationale for the bridging rules outside the ontologies themselves, and in the usage made of the instances of concepts in these ontologies. As Borgida and Serafini observe, “much, though by no means all, work in information integration has assumed that the same individual appears in identical form in the different sources of information being integrated” [4]. Borgida and Serafini then study distributed description logics for some other cases, in which directional mappings are needed – e.g., as when converting from one currency to another and charging different surcharges depending on the direction of the conversion, or in the case of two databases, one carrying information about individuals, the other about married couples of individuals. Here, we clearly are in such a setting, where more subtle relationships hold between the individuals of the different domains. Consequently, we used heuristics – such as the strong agency relationship principle – to provide a rationale for – in other words, to justify – the bridging rules defined in this paper.
- *Methodologies for software engineering perspective:* One of the recurring difficulties in the design of software engineering methodologies is how to move between levels of abstraction. For instance, a first level of abstraction are requirements, the second one is the architecture, the third the behavior of individual components, another one is the organization of data, and so on. Moving from one level of abstraction to the next (as well as going back) is – at best – guided by heuristics; more often, it is left to the intuition of the software engineer. The problem is that intuitions differ, while heuristics tend to be variously interpreted for better or for worse. Now, it is clear that all levels of abstraction involve implicit ontologies, which, in rare cases have been formalized. Also, the implicit ontologies behind each of the levels of abstraction are clearly different. It may then seem that heuristics are the best that software engineering can do at present. This paper argues and illustrates the opposite: if the ontologies are made explicit, heuristics can give rise to bridging rules, which in turn can be used to automate at least some of the activities performed when moving between levels of abstraction. Indeed, formalizing these ontologies requires effort, and obtaining bridging rules is not trivial, but it is a one-time investment. Once bridging rules are given, they can be spread and used at will, and given that they are explicit and their assumptions clear, they can be revised in a systematic manner, to give rise to more efficient, domain-specific revisions, or entirely new rules in case of new software engineering paradigms.

Future work is organized along the three perspectives identified above. In terms of norm-governed MAS, we are interested in validating the algorithms – for the specification of norms on the basis of a requirements specification – that arise from the bridging rules suggested here. In this respect, we will be combining the algorithms with one of the frameworks for the engineering of MAS mentioned in related work. From the applied ontology and software engineering perspectives, the question is whether we can apply the rationale of this paper to bridging other levels of abstraction and/or for other software engineering paradigms (e.g., requirements and architecture, or requirements and data). This will involve making explicit the core ontologies that are implicit at the different levels of abstraction, and establishing how the information newly added at each level relates to the information derived from the immediately preceding level of abstraction. From the software engineering perspective alone, the question is how available methodologies are affected by the availability of algorithms that automate tasks involved in moving between levels of abstraction.

## References

- [1] G. Y. Bizer, J. C. Barden, and R. E. Petty. Attitudes. In *Encyclopedia of Cognitive Science*. MacMillan, 2003.
- [2] Barry W. Boehm, Alexander Egyed, Julie Kwan, Daniel Port, Archita Shah, and Raymond J. Madachy. Using the winwin spiral model: A case study. *IEEE Computer*, 31(7):33–44, 1998.
- [3] Guido Boella and Leendert W. N. van der Torre. Regulative and constitutive norms in normative multi-agent systems. In *Principles of Knowledge Representation and Reasoning (KR)*, pages 255–266, 2004.
- [4] Alexander Borgida and Luciano Serafini. Distributed description logics: Assimilating information from peer sources. *J. Data Semantics*, 1:153–184, 2003.
- [5] Joost Breuker and Rinke Hoekstra. Core concepts of law: taking common-sense seriously. In *Proceedings of Formal Ontologies in Information Systems FOIS-2004*, 2004.
- [6] Joost Breuker, Rinke Hoekstra, Alexander Boer, Kasper van den Berg, Rossella Rubino, Giovanni Sartor, Monica Palmirani, Adam Wyner, and Trevor Bench-Capon. Deliverable 1.4, Estrella. Owl ontology of basic legal concepts (Iklif-core) – deliverable 1.4. Technical report, The European project for Standardized Transparent Representations in order to Extend Legal Accessibility (Estrella), 2007.
- [7] Joost Breuker, Andre Valente, and Radboud Winkels. Use and reuse of legal ontologies in knowledge engineering and information management. In V.R. Benjamins, P. Casanovas, J. Breuker, and A. Gangemi, editors, *Law and the Semantic Web*. Springer, 2004.
- [8] Jan Broersen, Mehdi Dastani, Joris Hulstijn, Zhisheng Huang, and Leendert W. N. van der Torre. The boid architecture: conflicts between beliefs, obligations, intentions and desires. In *Agents*, pages 9–16, 2001.
- [9] P. Casanovas, N. Casellas J.-J. Vallbé, M. Poblet, V. R. Benjamins, M. Blázquez, R. Pe na, and J. Contreras. Semantic web: A legal case study. In *Semantic Web Technologies*. Wiley, 2006.
- [10] Cristiano Castelfranchi, Frank Dignum, Catholijn M. Jonker, and Jan Treur. Deliberative normative agents: Principles and architecture. In *ATAL*, pages 364–378, 1999.
- [11] Jaelson Castro, Manuel Kolp, and John Mylopoulos. Towards requirements-driven information systems engineering: the tropos project. *Inf. Syst.*, 27(6):365–389, 2002.
- [12] Philipp Cimiano, Andreas Eberhart, Pascal Hitzler, Daniel Oberle, Steffen Staab, and Rudi Studer. The smartweb foundational ontology. Technical report, SmartWeb Project Report, SEP 2004.
- [13] Philip R. Cohen and Hector J. Levesque. Intention is choice with commitment. *Artif. Intell.*, 42(2-3):213–261, 1990.
- [14] R. Conte and C. Castelfranchi. *Cognitive and Social Action*. UCL Press, 1995.
- [15] Rosaria Conte and Cristiano Castelfranchi. Norms as mental objects - from normative beliefs to normative goals. In *From Reaction to Cognition, 5th European Workshop on Modelling Autonomous Agents, MAAMAW*, pages 186–196, 1993.
- [16] Cycorp. Opencyc.

- [17] Anne Dardenne, Axel van Lamsweerde, and Stephen Fickas. Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1-2):3–50, 1993.
- [18] Robert Darimont and Axel van Lamsweerde. Formal refinement patterns for goal-driven requirements elaboration. In *SIGSOFT FSE*, pages 179–190, 1996.
- [19] Nimit Desai, Amit K. Chopra, and Munindar P. Singh. Business process adaptations via protocols. In *IEEE International Conference on Services Computing (SCC)*, pages 103–110, 2006.
- [20] M. d’Inverno and M. Luck. *Understanding Agent Systems*. Springer-Verlag, 2001.
- [21] A. Eagly and S. Chaiken. Attitude structure and function. In *The Handbook of Social Psychology (4th ed.)*. New York: McGraw-Hill, 1996.
- [22] Marc Esteva, Julian A. Padget, and Carles Sierra. Formalizing a language for institutions and norms. In *ATAL*, pages 348–366, 2001.
- [23] R. Ferrario and A. Oltramari. Towards a computational ontology of mind. In *Proceedings of Formal Ontologies in Information Systems FOIS-2004*, 2004.
- [24] Ariel Fuxman, Lin Liu, John Mylopoulos, Marco Roveri, and Paolo Traverso. Specifying and analyzing early requirements in tropos. *Requir. Eng.*, 9(2):132–150, 2004.
- [25] Aldo Gangemi and Peter Mika. Understanding the semantic web through descriptions and situations. In *CoopIS/DOA/ODBASE*, pages 689–706, 2003.
- [26] Peter Gärdenfors. *Conceptual Spaces: the Geometry of Thought*. Cambridge, MA: MIT Press, 2000.
- [27] Rinke Hoekstra, Joost Breuker, Marcello Di Bello, and Alexander Boer. The IkiF core ontology of basic legal concepts. In *Proceedings of the Workshop on Legal Ontologies and Artificial Intelligence Techniques (LOAIT)*, 2007.
- [28] Wesley Newcomb Hohfeld. *Fundamental Legal Conceptions as Applied in Legal Reasoning*. Yale University Press, 1919.
- [29] IEEE Standard Upper Ontology Working Group. Sumo (suggested upper merged ontology).
- [30] Nicholas R. Jennings. On agent-based software engineering. *Artif. Intell.*, 117(2):277–296, 2000.
- [31] Nicholas R. Jennings, Timothy J. Norman, Peyman Faratin, P. O’Brien, and Brian Odgers. Autonomous agents for business process management. *Applied Artificial Intelligence*, 14(2):145–189, 2000.
- [32] Nicholas R. Jennings, Katia P. Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.
- [33] Michael C. Jensen and William H. Meckling. Theory of the firm: Managerial behavior, agency costs and ownership structure. *J. Financial Economics*, 3(4):305–360, 1976.
- [34] Ivan J. Jureta, Stéphane Faulkner, and Pierre-Yves Schobbens. A more expressive softgoal conceptualization for quality requirements analysis. In *Proceedings of the 25th International Conference on Conceptual Modelling (ER’06)*, 2006.
- [35] Ivan J. Jureta, Stéphane Faulkner, and Pierre-Yves Schobbens. Achieving, satisficing, excelling. In *Proceedings of the 1st International Workshop on Requirements, Intentions and Goals in Conceptual Modeling (RIGiM’07)*, 2007.
- [36] Ivan J. Jureta, Martin Kollingbaum, Stéphane Faulkner, John Mylopoulos, and Katia Sycara. Requirements-driven contracting for norm-governed multiagent systems. Technical report, Available online: <http://www.jureta.net/papers/RDC.pdf>, 2007.
- [37] Ivan J. Jureta, John Mylopoulos, and Stéphane Faulkner. Revisiting the core ontology and problem in requirements engineering. Technical report, Available online: <http://www.jureta.net/papers/core.pdf>, 2008.
- [38] D. Kahneman, I. Ritov, and D. Schkade. Economic preferences or attitude expressions?: An analysis of dollar responses to public issues. *J. Risk and Uncertainty*, 19, 1999.
- [39] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: The state of the art. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2005. <<http://drops.dagstuhl.de/opus/volltexte/2005/40>>.
- [40] Martin Kollingbaum, Ivan J. Jureta, Wamberto Vasconcelos, and Katia Sycara. Automated requirements-driven definition of norms for the regulation of behavior in multi-agent systems. In *Proceedings of the Symposium on Behaviour Regulation in Multi-Agent Systems (BRMAS)*, 2008.
- [41] Martin J. Kollingbaum. *Norm-Governed Practical Reasoning Agents*. PhD thesis, Dept. of Computing Science, University of Aberdeen, 2005.
- [42] Martin J. Kollingbaum, Wamberto Weber Vasconcelos, Andrés García-Camino, and Timothy J. Norman.

- Conflict resolution in norm-regulated environments via unification and constraints. In *Declarative Agent Languages and Technologies V, 5th International Workshop, DALT 2007, Honolulu, HI, USA, May 14, 2007, Revised Selected and Invited Papers*, pages 158–174, 2007.
- [43] Manuel Kolp, Paolo Giorgini, and John Mylopoulos. Multi-agent architectures as organizational structures. *Autonomous Agents and Multi-Agent Systems*, 13(1):3–25, 2006.
- [44] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltamari, and L. Schneider. DOLCE : a Descriptive Ontology for Linguistic and Cognitive Engineering. Technical report, Institute of Cognitive Science and Technology, Italian National Research Council, 2003.
- [45] N. H. Minsky and V. Ungureanu. A mechanism for establishing policies for electronic commerce. In *ICDCS '98: Proceedings of the The 18th International Conference on Distributed Computing Systems*, page 322, Washington, DC, USA, 1998. IEEE Computer Society.
- [46] Naftaly H. Minsky and Victoria Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *ACM Trans. Softw. Eng. Methodol.*, 9(3):273–305, 2000.
- [47] J. Mylopoulos, L. Chung, and B. Nixon. Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Trans. Softw. Eng.*, 18(6):483–497, 1992.
- [48] Timothy J. Norman, Alun D. Preece, Stuart Chalmers, Nicholas R. Jennings, Michael Luck, Viet Dung Dang, Thuc Duong Nguyen, Vikas Deora, Jianhua Shao, W. A. Gray, and N. J. Fiddian. Agent-based formation of virtual organisations. *Knowl.-Based Syst.*, 17(2-4):103–111, 2004.
- [49] Daniel E. O’Leary, Daniel Kuokka, and Robert Plant. Artificial intelligence and virtual organizations. *Commun. ACM*, 40(1):52–59, 1997.
- [50] Andrea Omicini. Soda: Societies and infrastructures in the analysis and design of agent-based systems. In *Proceedings of the International Workshop on Agent-Oriented Software Engineering (AOSE)*, pages 185–193, 2000.
- [51] G. Oppy. Propositional attitudes. In *Routledge Encyclopedia of Philosophy*. London: Routledge, 1998.
- [52] Olga Pacheco and José Carmo. A role based model for the normative specification of organized collective agency and agents interaction. *Autonomous Agents and Multi-Agent Systems*, 6(2):145–184, 2003.
- [53] P. F. Patel-Schneider and I. Horrocks. Owl web ontology language semantics and abstract syntax. Technical report, W3C, 2004.
- [54] Luc Schneider. Designing foundational ontologies: The object-centered high-level reference ontology ochre as a case study. In *Proceedings of the International Conference on Conceptual Modeling (ER)*, pages 91–104, 2003.
- [55] J. R. Searle. *Speech acts: An essay in the philosophy of language*. Cambridge: Cambridge University Press, 1969.
- [56] J. R. Searle. A taxonomy of illocutionary acts. In *Language, Mind, and Knowledge (Studies in the Philosophy of Science, Vol.7)*. Minneapolis: University of Minnesota Press, 1975.
- [57] Y. Shoham and M. Tennenholz. On social laws for artificial agent societies: Offline design. In *Computational Theories of Interaction and Agency*. MIT Press, 1996.
- [58] Barry Smith and Pierre Grenon. Basic formal ontology (bfo). INFOMIS Reports.
- [59] André Valente, Joost Breuker, and Bob Brouwer. Legal modeling and automated reasoning with on-line. *Int. J. Hum.-Comput. Stud.*, 51(6):1079–1125, 1999.
- [60] Axel van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *5th IEEE International Symposium on Requirements Engineering (RE 2001), 27-31 August 2001, Toronto, Canada*, page 249, 2001.
- [61] Axel van Lamsweerde, Robert Darimont, and Emmanuel Letier. Managing conflicts in goal-driven requirements engineering. *IEEE Trans. Software Eng.*, 24(11):908–926, 1998.
- [62] Wamberto Weber Vasconcelos, Martin J. Kollingbaum, and Timothy J. Norman. Resolving conflict and inconsistency in norm-regulated virtual organizations. In *6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007), Honolulu, Hawaii, USA, May 14-18, 2007*, page 91, 2007.
- [63] Javier Vázquez-Salceda, Virginia Dignum, and Frank Dignum. Organizing multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 11(3):307–360, 2005.
- [64] Fabiola López y López, Michael Luck, and Mark d’Inverno. Constraining autonomy through norms. In *AAMAS*, pages 674–681, 2002.
- [65] Fabiola Lopez y Lopez, Michael Luck, and Mark d’Inverno. A normative framework for agent-based systems. In Guido Boella, Leon van der Torre, and Harko Verhagen, editors,

*Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2007. <<http://drops.dagstuhl.de/opus/volltexte/2007/933>>.

- [66] Eric Yu. Towards modeling and reasoning support for early requirements engineering. In *Proceedings of the IEEE International Symposium on Requirements Engineering*, 1997.
- [67] Franco Zambonelli, Nicholas R. Jennings, and Michael Wooldridge. Developing multiagent systems: The gaia methodology. *ACM Trans. Softw. Eng. Methodol.*, 12(3):317–370, 2003.