The Knowledge Engineering Review, Vol. 30:5, 564–606.
doi:10.1017/S0269888915000065

Computational logics and verification techniques of multi-agent commitments: survey

MOHAMED EL MENSHAWY[1,2], JAMAL BENTAHAR[1], WARDA EL KHOLY[1], PINAR YOLUM[3] and RACHIDA DSSOULI[1]

[1]*Faculty of Engineering and Computer Science, Concordia University, Montreal, Canada;*
e-mail: m_elme@encs.concordia.ca, bentahar@ciise.concordia.ca, moh_marzok75@yahoo.com, w_elkh@encs.concordia.ca, rachida.dssouli@concordia.ca;
[2]*Faculty of Computers and Information, Menofia University, Shebin El Kom, Egypt;*
e-mail: m_elme@encs.concordia.ca;
[3]*Department of Computer Engineering, Bogazici University, Istanbul, Turkey;*
e-mail: pinar.yolum@boun.edu.tr

Abstract

Agent communication languages (ACLs) are fundamental mechanisms that enable agents in multi-agent systems to *talk*, communicate with each other in order to satisfy their individual and social goals in a cooperative and competitive manner. Social approaches are advocated to overcome the shortcomings of ACL semantics delineated by using mental approaches in the figure of agents' mental notions. Over the last two decades, *social* commitments have been the subject of considerable research in some of those social approaches as they provide a powerful representation for modeling and reasoning upon multi-agent interactions in the form of mutual contractual obligations. They particularly provide a declarative, flexible, verifiable, and social semantics for ACL messages while respecting agents' autonomy, heterogeneity, and openness.

In this manuscript, we go through prominent and predominate proposals in the literature to explore the state of the art on how temporal logics can be devoted to define a formal semantics for ACL messages in terms of social commitments and associated actions. We explain each proposal and point out if and how it meets *seven* crucial criteria, four of them introduced by Munindar P. Singh to have a well-defined semantics for ACL messages. Far from deciding the best proposal, our aim is to present the advantages (strengths) and limitations of those proposals to designers and developers using a concrete running example and to compare between them, so that they can make the best choice with regard to their needs. We explore and evaluate current specification languages and different verification techniques that have been discussed within those proposals to, respectively, specify and verify commitment-based protocols. We also investigate logical languages of actions advocated to specify, model, and execute commitment-based protocols in other contributed proposals. Finally, we suggest some solutions that can contribute to address the identified limitations.

1 Introduction

Communication among autonomous and heterogeneous agents is a fundamental aspect for building open and *effective* multi-agent systems (MASs). The autonomy property means that agents are free to communicate as they are delighting, but behave with respect to the rationale of their state of affairs, plans, and goals. The heterogeneity property means that agents are designed and implemented in different ways (i.e. agents are following different and probably opposite goals, desires, and intentions). In contrary, homogeneous agents have the same capabilities, as they are designed in an identical way. The reason that makes communication significant is not only because agents have the property of social ability and

because our ultimate objective is in turn to simulate *human societies* where communication is quintessential, but also because dynamic systems develop through communication among participating entities. In such systems, entities must communicate to negotiate deals (e.g. price of goods, delivery terms, or payment conditions), cooperate, and even compete with each other to satisfy their individual and social goals, which they cannot achieve alone (due to the lack of resources, capabilities, or knowledge) (Weiss, 1999). Another crucial reason is that today's applications of MASs ranging from agent-based Web services and their communities (Bentahar *et al*., 2008), human learners (Sklar & Richards, 2010), software agent-based interaction models (Cabri *et al.*, 2010), robotic agents (Nakano *et al.*, 2011; Vidoni *et al.*, 2011), multi-sensory and distributed surveillance (Gascueña & Fernández-Caballero, 2011), agent-based computational economic models (Richiardia, 2012), manufacturing systems (Lim & Zhang, 2012) to Web service composition (Lomuscio *et al.*, 2012), have one thing in common: The agents employed in those systems should communicate with each other. It is clear that the success of those MASs requires commonly understood languages: *lingua franca* for agents to talk to each other in order to determine what proper information to exchange or what right action to take as well as powerful mechanisms (protocols) to regulate and coordinate communication among participants within conversations.

1.1 Mental approaches

A promising method in fact to model agent communication was widely inspired by John Searle's *speech act theory* (Searle, 1969). This theory is also named *communicative act theory* (Wooldridge, 2009) due to the philosopher of language, John Austin, where linguistic communication is a special type of actions, which interacting agents can perform in an attempt to appropriately influence other agents (Weiss, 1999; Wooldridge, 2009). In this theory, as agents are autonomous, there is no choice for an agent, except in principle to affect another agent or similarly be influenced by another agent by carrying out an action. The following is a famous example in agent communication literature.

Example 1 *Once a judge announces a couple married, the declaration utterance changes the 'state of the world in a very tangible way' (Wooldridge, 2009), which in turn means that the judge is publicly bringing the known fact into an existence state.*

The communicative act theory particularly pertained to identify actions—denoted by *performative* verbs and also mapped into agents messages—performed by agents to satisfy a rational balance between their mental states (also named cognitive notions) such as *beliefs*, *goals*, *desires*, and *intentions*. This is particularly named *mental approach*. For instance, a mental approach specifies how the agent's beliefs update as a consequence of the occurrence of a definite action. Mental approaches were exploited and being utilized to define the semantics of agent communication languages (ACL) messages employed in the most popular and standard ACLs, such as Knowledge Query and Manipulation Language (KQML) (Labrou & Finin, 1998) and ACLs of the foundation for intelligent physical agents (FIPA-ACL)[1]. Technically, the formal semantics of FIPA-ACL messages were given with respect to a well-defined formal language called Semantic Language (SL). FIPA-SL[2] is a quantified multi-modal logic with several referential operators (e.g. *any*, *all*) and actions (e.g. *feasible*, *done*). The idea of SL semantics is to map each FIPA-ACL message into a formula of SL, which in turn defines a constraint that the sender of the message must achieve before uttering the message's performative. Such a constraint is so-called the *feasibility precondition*. The SL semantics also map the postcondition of the message into the *rational effect* of the action: What an agent will be attempting to satisfy by performing the message's performative.

Example 2 *The FIPA-SL semantics of the performative inform is defined as*:

$\langle i, inform(j, \varphi)\rangle$

feasibility precondition: $B_{i\varphi} \wedge \neg B_i(Bf_{i\varphi} \vee Uf_{i\varphi})$

rational effect: $B_i\varphi$

[1] FIPA-ACL message structure specification, `http://www.fipa.org/specs/fipa00061/`

[2] FIPA-SL content language specification, `http://www.fipa.org/specs/fipa00008/index.html`

The SL expressions $B_i\varphi$, $Bf_i\varphi$, and $Uf_i\varphi$ mean, respectively, that agent i believes φ, agent j has a certain way about the truth or falsity of φ, and agent j is uncertain about φ. Thus, agent i informing agent j that φ is true means that i believes φ and it does not believe that either j believes φ is true or false, or j is uncertain about the truth or falsity of φ. The rational effect is that the receiver agent j will believe φ.

While cognitive notions are desirable abstractions for designing *individual* agents, an ACL should also support the interoperability of two or more participating agents (Singh, 1998, 2000). As known, the interoperability is the capability of heterogeneous agents to work with each other to build effective and feasible systems. To satisfy the interoperability aspect, an agent should be in principle able to compute and infer an abstraction of the mental (or internal) state of another agent. Such a principle is not provided in the mental semantics (Singh, 1998; Chopra *et al.*, 2013). Furthermore, it became apparent that a purely mental semantics for ACL messages in terms of pre- and post-conditions could inevitably impose substantial restrictions on the functional behaviors of agents.

Example 3 *Consider a business scenario where a merchant agent informs a customer agent that a shipment will deliver on Monday.*

When the shipment basically fails to deliver on Monday, then there is no way for the customer to verify whether or not the merchant sincerely believes that the shipment should be delivered on Monday, delineated by the inform's precondition (Singh, 1998, 2000). Initially, the merchant could be particularly deceiving. On the other hand, the customer would never know whether or not the merchant would conform to ship goods without an external observer that can access the merchant's database (it is like reading the merchant's mind). Such an observer can be developed and implemented in specific situations, but the notion in principle is far from the standard in real business domains, because participating agents will not be willing to reveal their internal functionalities (Singh, 1998; Chopra *et al.*, 2013). This problem is known as the *semantics verification* problem (Wooldridge, 2000, 2009). Accordingly, such pure mental semantics would be of severely restricted value in systems of heterogeneous agents (Singh, 2000).

When conversing, agents do not particularly exchange separated messages, but a succession of messages. The developers of FIPA-ACL, for instance, have addressed the challenge of integrating ACLs and protocols by proposing a set of *conversation policies*, named FIPA-ACL protocols[3]. These protocols involve messages of the standard types in FIPA and can be viewed as specific ACLs designed for particular purposes, such as Request Interaction protocol, English Auction Interaction protocol (EAI), and Contract Net protocol (CN). For examples, the CN protocol is designed from online business's perspective to reach agreements among interacting agents, whereas the EAI protocol is designed to help auction goods acquire a higher market value. FIPA-ACL protocols have succeeded and exploited in practice to specify the rules governing interactions and coordinating dialogues among participating agents by: (1) curbing the range of permitted follow-up messages at any phase during a dialogue; and (2) describing the possible ordering and occurrence constraints on messages that FIPA compliant agents can exchange for particular applications. Such protocols, however, are too rigid to be utilized by autonomous agents. This is because they are specified so that agents should enact them without being able to handle exceptions that emerge at run time. This restricts obviously the protocols' flexibility (El-Menshawy *et al.*, 2011b; Chopra *et al.*, 2013).

We do not intend to in fact supply a comprehensive review of all current approaches introduced to address the limitations of mental approaches and FIPA-ACL protocols as this would be beyond the scope of this article. Therefore, we refer interested readers, for example, to: (1) the survey introduced by Maudet and Chaib-Draa (2002), which in turn focuses on how dialogue games can be exploited to improve the flexibility of FIPA-ACL protocols; (2) the book *Issues in Agent Communication* edited by Dignum and Greaves (2000), which documents two workshops on communication in MAS held in 1999: one on 'Specifying and Implementing Conversation Policies' and the other on 'ACLs'; and (3) the recent manifesto presented by agent communication researchers (Chopra *et al.*, 2013), which in turn aims at identifying consensus and substantial differences among current approaches and interesting directions of future work in the field of agent communication. In the following, we only focus on social commitments and commitment-based protocols.

[3] See FIPA-ACL interaction protocols (2001, 2002), `http://www.fipa.org/repository/ips.php3`

1.2 Social approaches

To overcome the shortcomings and inconveniences incorporated with mental approaches, the MAS community witnessed a shift from individual agent representations to social interaction approaches (Singh, 1998, 2000) by particularly abstracting away from agents' mental states and by providing a social meaning to agent message exchanges. From Example 1, the judge might sign and then affix his seal on a marriage certificate (Wooldridge, 2000) or might simply talk in public, because what specifically induces the truth communication is the agreed convention within a given business domain. Informally, we would think of the judge's proclamation as saying 'I declare this couple husband and wife'. Parallel with this thinking, Austin argued that all communications could be articulated in the *declarative* form by making use of appropriate performative verbs (Wooldridge, 2000, 2009). Thus, from Example 3, an *informative* verb such as 'the shipment will deliver on Monday' can be treated as 'merchant informs customer that the shipment will deliver on Monday'. Furthermore, a *directive* verb such as 'send the goods to customer' can be treated as 'customer requests that merchant sends the required goods to him'. In fact, such a shift is recently asserted by agent communication community in a study by Chopra *et al.* (2013). Commitments are employed in some of these social approaches, which successfully supply a well-founded principle for representing, modeling, and reasoning upon the communication of agents. In Longman Dictionary, commitment is:

- 'a promise to do something or to behave in a particular way'; or
- 'something that you have promised you will do or that you have to do'.

In such a definition, commitment imposes loyalty, dedication, or devotion toward a person within a social community. In broad terms, commitments are social, objective, and help in representing the states of affairs at different instants in the course of multi-agent interactions. Informally, social commitment expresses an engagement made by an agent, named the debtor—as a result of messages exchanged, actions taken, or other types of events related to agent communication—and is directed toward another agent, named the creditor, to bring about certain fact (Singh, 1999) or to achieve some consequence of interest. For what concerns, the notion of social commitments briefly provides the following benefits for agent communication.

- The main idea of social semantics is to solely define public aspects of ACL messages for further reference from high-level abstractions without entirely reasoning about agent's mental states. That is, all social states of the MAS that contain social facts such as commitments are global, accessible, and shared by the pairs of agents that are concerned with these social facts so that (1) the satisfiability (soundness) (Fornara & Colombetti, 2003) of agents' behaviors can be easily monitored and verified (Chesani *et al.*, 2013; Spoletini & Verdicchio, 2007, 2009; Torroni *et al.*, 2010); and (2) the interoperability among interacting agents operating in heterogeneous systems can be straightforwardly achieved (Chopra & Singh, 2008; Baldoni *et al.*, 2010).
- Commitment-based approaches accommodate the autonomy and flexibility of the interacting agents. The autonomy aspect is satisfied in a natural way as each agent is in principle expected to only achieve its commitments. On the other hand, as an agent at run time can select what is best for satisfying its goals (e.g. a customer substitutes a new method to make a payment, selects to deliver first, or returns damaged goods), then the approach supports flexibility as long as the behavior is correct at the business interaction level (Yolum & Singh, 2004; Desai *et al.*, 2005; Winikoff *et al.*, 2005; Chopra *et al.*, 2013).
- Commitment-based approaches provide a robust way to characterize the degrees of autonomy and interdependency of interacting agents without considering low-level details (Chopra *et al.*, 2013). On the one hand, when the interacting agents are completely autonomous, we then will not have an effective system of agents. On the other hand, when there is no interdependence, the interacting agents will be approximately useless; instead autonomous agents must be able to cooperate and compete with each other.
- An important aspect of social commitments is that commitments can be modified and manipulated using a set of named commitment actions, such as *Create, Discharge*, and *Cancel* (Singh, 1999; Venkatraman & Singh, 1999). More precisely, such actions capture the dynamic behavior of committing agents,

provide a principle way to evolve the changes in the social commitment states, and define the life cycle of commitments.

- As social commitments have been introduced in agent communication research, they have become a well-acknowledged engineering tool to formally model and reason upon the interaction among agents (Yolum & Singh, 2002b, 2004). Furthermore, they have been successfully applied to other research areas such as modeling business processes (Desai *et al.*, 2005; Telang & Singh, 2009a, 2012), developing artificial institutions (Fornara *et al.*, 2008), defining programming languages (Winikoff, 2007; Singh & Chopra, 2010), modeling service-oriented computing (Singh *et al.*, 2009), enhancing agent-oriented software engineering methodologies (El-Menshawy *et al.*, 2009a; Telang & Singh, 2009b), developing Web-based MASs (Venkatraman & Singh, 1999), developing agent-based Web services and their communities (Bentahar *et al.*, 2008; El-Menshawy *et al.*, 2009a), specifying commitment-based protocols (Yolum & Singh, 2002b, 2004; Desai *et al.*, 2007; Mallya & Singh, 2007; Baldoni *et al.*, 2010), specifying business protocols (Desai *et al.*, 2005; El-Menshawy *et al.*, 2010a), checking commitment conflicts (Günay &Yolum, 2013), and diagnosing exceptions (Kafali & Torroni, 2012).

Current proposals considered defining the formal semantics of ACL messages in terms of social commitments by means of either temporal logics[4] (Singh, 2000; Bentahar *et al.*, 2003, 2004b, 2007; Mallya & Huhns, 2003; Mallya *et al.*, 2004; El-Menshawy *et al.*, 2010b), which in turn we term 'logics of commitments', or logics of actions (Yolum & Singh, 2002b, 2004; Chopra & Singh, 2006; Desai & Singh, 2007; Chesani *et al.*, 2009). Through this article, we call these two types of logics, *computational logics of commitments*. The main advantages of using computational logics can be summarized as follows:

- They have a well-understood formal semantics, which in turn lays down the foundation for a neat, concise, and unambiguous logical meaning of agent messages.
- They provide a high-level language in which notions can be expressed in a transparent and concise way through a small range of powerful constructs.
- They allow us to model not only static aspects of an agent state, but also the dynamic behavior of agents. More significantly, they can be exploited by agents for making decisions and selecting courses of action, and they provide explanations for conclusions in a human friendly way (i.e. the sequence of inference steps would be restored).
- They facilitate and improve the applicability of the proposed semantics to be exploited, for example to develop automated reasoners.

1.3 Specifying and verifying commitment-based protocols

Commitment-based protocols, specified as a set of commitments and associated actions explicitly capturing the meanings of exchanged messages and a set of agent roles, are more suitable for regulating and coordinating agent interactions than FIPA-ACL's protocols and conventional software engineering protocols purely modeled in terms of *finite state machines* (FSM), *statecharts*, and *Petri nets*, which only capture legal orderings of exchanged messages without unduly considering the meanings of those messages (El-Menshawy *et al.*, 2011b; Chopra *et al.*, 2013). In a point of fact, the typical function of commitments in commitment-based protocols involves introducing the syntax for the exchanged messages along with a formalization of the precise meanings (expressed in terms of commitments) of those messages. Having such a declarative principle not only simplifies the modeling of protocols, but also provides a beneficial basis for flexible and rigorous specifications (i.e. a participating agent might communicate in any way it opts as long as its created commitments are discharged), which in turn can be searched for correctness (Yolum & Singh, 2004; Yolum, 2007; Bentahar *et al.*, 2010; El-Menshawy *et al.*, 2010b, 2011b). Internally, in commitment-based protocols, agents will not reason about legal sequences but about concrete commitment states and possible paths[5] to reach them (Yolum & Singh, 2002a; El-Menshawy *et al.*, 2011c).

[4] Temporal logic, a special case of modal logic, consents to reason about temporal relations and qualitative aspects of time in an abstract sense.

[5] A computation path is an infinite sequence of system's states.

The motivation of the MAS community, however, is no longer just formally representing and reasoning about social commitments and commitment actions, but becomes the application of various verification techniques, such as full-automatic model checking, semi-automatic verification, local testing, static verification, monitoring, etc. in order to (1) verify the compliance of commitment-based protocols with given specifications at design time; (2) identify the compliant and non-compliant agents at the end of the commitment-based protocol at run time; or (3) track the evolution of commitment statuses at run time, respectively. This verification process is significant because there is a possibility that an agent might fail to comply with its commitments as it is supposed to. For what concerns, it is unrealistic—in open systems (e.g. e-business, e-negotiation) or in any application wherein interacting agents are implemented in different programming languages and having competing objectives—to assume that all autonomous agents will behave (or act) according to the given protocols because they might not behave as they are committed to or at least not wantonly violate or cancel commitments. Furthermore, a formal verification technique, such as model checking, is necessary to help protocol designers either detect automatically unwanted and bad agents' behaviors to eliminate them or enforce desirable agents' behaviors so that such protocols comply with given specifications. Thus, ensuring that merely the suitable interactions take place is a significant aspect of analyzing and designing effective MASs (El-Menshawy *et al.*, 2010b, 2011c).

1.4 Motivation and organization of the article

The motivation and plan of the article are to go through prominent and predominate proposals in the literature to explore the state of the art on how temporal logics can be devoted to define a formal semantics for ACL messages in terms of social commitments and related actions. In our methodology to achieve this objective, we first begin by exploring the historical development of commitments from philosophy, distributed systems, and artificial intelligence (AI) perspectives up to the time it gets landed on MASs, including as a special case agent communication (Section 2). Second, in Section 3, we present a running example to demonstrate our presentation and discussion of reviewed proposals. Third, in Sections 4–6, we explore and evaluate current proposals that advocate temporal logics: respectively, *linear temporal logic* (LTL) (Pnueli, 1977), *computation tree logic* (CTL) (Clarke & Emerson, 1982), and *full computation tree logic* (CTL*) (Emerson & Halpern, 1986) not only to represent and reason about commitments and associated actions, but also to specify commitment-based protocols. We specifically explain each proposal and point out if and how it meets seven crucial criteria (presented in Section 2). We also aim at exploring and evaluating different techniques proposed to verify the correctness of commitment-based protocol specifications introduced in those proposals. The point is not to declare one proposal as a winner, but to highlight the advantages (strengths) and limitations (where they exist) of those proposals to designers and developers, so that they can make the best choice with regard to their needs. The limitations (or disadvantages) should not be considered fatal, but only a consideration that must be taken. Fourth, in Section 7, we proceed to succinctly present logical languages of actions, such as event calculus (EC), C^+, and modular action description for protocols, which have been exploited to specify, model, and execute commitment-based protocols. In Section 8, we finally conclude by summarizing current limitations and suggesting the directions for future work from our perspective. It is worth noticing that our methodology would help designers and developers with their choices throughout:

1. Indicating that CTL*-based approaches are more expressive than those using LTL and CTL, which are in turn incomparable, expressiveness-wise (Emerson, 1990). This will help designers, who are non-expert in logic, make their choices if expressiveness is an issue.
2. Presenting the syntax and semantics of commitments and their fulfillments to be able to express them in the desirable properties.
3. Discussing the way of defining the formal models introduced in the reviewed proposals and the way of expressing desirable protocol properties using LTL, CTL, and CTL*.
4. Categorizing current proposals into run-time proposals and design time proposals with respect to the employed verification methods.
5. Comparing different verification techniques in terms of their efficiency: The number of interacting agents, number of reachable states, and execution time.
6. Presenting our conclusions about each section in the form of remarks.

We finally present a running example, which we use in our presentation and discussion of reviewed proposals through the whole manuscript.

2 Bringing history to commitments

In this section, we show that commitments in MASs and agent communication differ from the ones having been discussed in other fields, such as philosophy, AI, and distributed systems.

In philosophy, the notion of commitment to a proposition goes back to at least the philosopher Hamblin (1970). In particular, Hamblin introduced the concept of *commitment store* as a public repository for storing all commitments that have been made by each participant in the dialogue. Hamblin's notion has been further developed in contributions to dialogue games that in turn cater a constructive proof-theory for propositions in a classical logic by the philosophers Walton and Krabbe (1995). In Walton and Krabbe's model, each dialogue game in principle comprises of five basic components, one of them is named *commitment rules*, which in turn define the conditions under which participating agents can bring their commitments to certain propositions. These commitments reflect the idea that the debtors are accepting a claim as an assertion about the truth of those propositions.

In the AI planning literature, Sacerdoti (1977) advocated an approach called *least commitment* planning. This commitment is a psychological one as it is a state of mind. A planner, an essential part of a single agent, creates plans that enable deferring decisions as only the partial ordering decisions are recorded. Technically, a key aspect of least commitment strategy is keeping the track of past decisions and their reasons. For example, if you purchase plane tickets (to satisfy the goal of boarding the plane), you should be sure to take them to the airdrome. When another goal (say, having your hands free to be able to open the taxi door) causes you to drop down the tickets, then you would be sure to perceive them again. This is in turn a fine notion for a single agent; however, when least commitments are canceled or stop work for any reasons, an agent will not be able to carry out any decision, that is, it becomes useless.

In distributed systems, mutual commit protocols ensuring that a number of distributed processes in principle agree upon some important actions are widely recognized. The most famous of those protocols is named *two-phase commit* (2PC) (Gray, 1978). 2PC uses a single coordinator to reach agreement. In the phase of commit-request, each process simply votes Yes or No to perform some required changes. In the commit-outcome phase, these votes are collected by the coordinator. When all processes voted Yes, the coordinator in turn announces a Yes decision. Otherwise a No decision is declared publicly. This represents some kind of commitment as the processes will behave according to the decision of the coordinator. Technically, representing a commitment as a 'flag' inside each process is, in principle, quite simple, but it is in turn inflexible, irrevocable, and suitable only for a single agent. Jennings (1993) presented commitments as a fundamental notion for efficient coordination in distributed systems. He also exploited the notion of conventions, in principle, to monitor the commitments and in turn to define the conditions under which commitments ought to be reevaluated and to then specify the associated actions that should be undertaken in such situations. The author further reformulates different models of coordination in terms of commitments that in turn provide a high-level goal for which the agent must specifically find a suitable course of action. However, commitments are modeled using mental notions, specifically using agent's intentions, purely for the purposes of coordination.

In early 1990s, the notion of commitments was in turn exploited in Belief-Desire-Intention framework to understand agent's intentions by considering the relationship between agent's persistent goals and actions as stated in the *rational interaction theory* (Cohen & Levesque, 1990). This approach is well suited to specifically represent a level of joint intentions (equally mutual beliefs) in individual agent's architecture in order to continually utilize the mental notions. Suppose p is a proposition. A mutual belief notion between two agents in turn means that each agent believes p and each agent believes that the other agent believes p in a nested form. However, satisfying this notion is more difficult in practical applications. Such a limitation (i.e. modeling commitments as a mental notion) incentives Singh (1991), in principle, to differentiate between two kinds of commitments: *psychological* commitment (i.e. a commitment of 'one agent to itself') as it is in principle exploited in AI and *social* commitment (i.e. a commitment of one agent to another 'to do certain actions'). The author ended with the psychological commitment that is in turn a

very restricted form of commitment as it specifically provides unidirectional relationship. Once psychologically committed to a particular intention or belief, an agent cannot consider it again, even the commitment conflicts its goals (Singh, 1996). In order to overcome this problem, the author argued that mutual beliefs are highly delicate and in principle hard to construct. Thus, psychological commitments and social commitments are different notions.

Singh's social notion of commitments has been further investigated by Castelfranchi (1995) to understand the social interactions among members of groups and organizations by introducing another agent, termed *witness* agent, in the context of social commitment. The witness agent in principle certifies the creation of commitment and plays a very crucial role in identifying cheater agents. Castelfranchi forcefully argued that the social relationship that engages interacting agents is, in principle, the key aspect that characterizes a commitment as being irreducible to shared knowledge or intentions. The author only focused on clarifying some concepts to be able in turn to propose a *descriptive ontology* to the theory of organizations without specifically considering the computational aspects of social commitments.

Castelfranchi's social and organizational metaphors provide a more straightforward way to think of MASs in terms of commitments. Singh (1997) started focusing solely on social commitment-based MAS approach to tackle distributed artificial intelligence (DAI) problems in heterogeneous and open information environments, termed *cooperative information systems*. He found out that the database approach for solving DAI problems is in principle too hard-wired and restrictive, whereas the agent approach specifically gives flexible solutions. The author also gave the definition of commitments using the approach called *spheres of commitments*, which in turn incorporates social policies to handle the creation, satisfaction, and cancelation actions of commitments and also relates commitments to organizational structures of MAS, in which agents can only acquire commitments after adopting their role through MAS execution.

Singh (1998) was the first to obviously stress the need for defining the semantics for ACLs in terms of social notions. The author pointed out the problem that none of the current ACLs (e.g. KQML) presupposes heterogeneous agents can communicate with each other not because of the lack of practice toward building a formal and public semantics that promotes social agency, but because the current practice is based solely upon mental agency, such as beliefs and intentions, which in turn failed to make agents autonomous and heterogeneous. He also argued that reading agents' minds is core to the mental approach, which is not the right direction, in principle, to deal with the problem. The author then presented design advantages of following the social approach over the mental one in terms of the meaning of ACL messages and agents constructions. Singh proceeded to show that the meaning of exchanged messages should be, in principle, characterized by the following seven communicative acts, namely *assertives*, *directives*, *permissives*, *expressives*, *commissives*, *declaratives*, and *prohibitives*.

Singh (1999) refined his seminal work introduced in Singh (1997) in order to unify normative concepts and commitments and hence coming up with a rich *descriptive ontology* of commitments that in turn generalizes Castelfranchi's social notion of groups (Castelfranchi, 1995). In this descriptive ontology, the relationship between commitments and norms is given by making use of 'meta-commitments'—a commitment about a commitment—to create a community and its norms. Singh delineated social unconditional commitment c by a four-argument relation involving a proposition p and three agents (i, j, and G): $C(i, j, G, p)$, which means that i is committed toward j in the organizational context G—which might be an agent or a system of agents such as eBay—to satisfy the proposition p. The agent i that actively makes the social commitment is named the *committer* (or debtor), the agent j to which the commitment is made is named the *committee* (or creditor), and p is named the *consequence* of the commitment. The context G includes the norms that apply to the group of agents wherein the commitment is established. Its main relational is to resolve disputes between the debtor and creditor, so unacceptable behaviors of the debtor and creditor can be managed. A debtor agent is free to determine which path to run so as to satisfy p. Recall that the debtor and creditor of the commitment are members of the social context. To make the analysis simpler, such a context has been removed from the notation of commitments in later proposals coming up from Singh and other researchers (a commitment c is denoted by $C(i, j, p)$). Singh elaborated other three actions, which are in turn exploited with the actions defined in Singh (1997) to manipulate commitments. As we understood, such manipulations provide a principle way to capture the changes in the social state of commitments. Chopra and Singh (2009) and El-Menshawy *et al.* (2010b) classified

those actions into: *two-* and *three-party* actions. The former actions (*Create*, *Discharge*, *Cancel* and *Release*) need only two agents to be performed and the latter action (*Delegate* and *Assign*) need an intermediate agent to be completed (see, e.g. Kafali & Torroni, 2011 for more details about the delegation action). In the following, we present the intended meaning of those actions and who has the right to perform them.

1. *Create* (i, c), performed by the debtor i to instantiate a new commitment c in the system and make the commitment active.
2. *Discharge* (i, c), when the commitment consequence is true, the commitment c is satisfied.
3. *Cancel* (i, c), performed by the debtor i to revoke its commitment c, which is no longer active.
4. *Release* (j, c), performed by the creditor j to free the debtor from carrying out its commitment c, which is no longer active.
5. *Delegate* (i, k, c), performed by the debtor i to shift its role to another debtor k, which, in principle, creates a new commitment $c' = C\ (k, j, p)$ in order to satisfy the current commitment on behalf of i.
6. *Assign* (j, k, c), performed by the present creditor j to transfer the current commitment to another creditor k, which, in principle, becomes the creditor of a new commitment $c' = C\ (i, k, p)$.

From El-Menshawy *et al.* (2010b), the state machine, that is the states wherein a social commitment can be held in and the state transitions, is so-called commitment life cycle. A commitment can be present in one state at a time and continues in that state till an action is carried out on it. Initially, the created commitment could be *Conditional* or *Unconditional* commitment. This is represented by the selection operator. A conditional commitment is denoted by $CC\ (i, j, p, q)$ and read as agent i commits to agent j to bring about the consequence q if the antecedent p is met. It can move either to *Negotiation* state to negotiate the antecedent of commitment among the participants or to *Antecedent* state to check the satisfaction of the antecedent. The conditional commitment could be negotiated many times until reaching either a mutual agreement about the antecedent, meaning that the negotiation's outcome is a conditional commitment where the antecedent is negotiated and agreed upon among the participants, or no agreement can be reached about the commitment antecedent wherein the commitment moves to the *Final* state. If the antecedent is not satisfied, the commitment moves to the *Final* state. Once the unconditional commitment is established, then it might either move to any one of the following states: *discharged*, *canceled*, *released*, *delegated*, *assigned,* or move to *negotiation* state again, but this time to negotiate the commitment consequence. When the participant agents reach an agreement, then the commitment moves to one of the aforementioned states or to the *Final* state if no agreement is being reached (Figure 1).

In Singh's (1999) approach , commitments and commitment actions are modeled, respectively, as *abstract objects* with names and *predicates*. For example, *Cancel* (i, c) denotes a predicate variable (i.e. a term in the predicate logic), which is true precisely when agent i cancels its commitment c. However, there is no formal semantics for commitments. Incorporating the notion of social commitments into agent communication and advocating it in principle to define the semantics for ACL messages was first introduced in early 2000s by Singh (2000). We return to Singh's approach in Section 5 along with a discussion of proposals that use CTL to define a formal semantics for ACL messages. A key strong point in Singh's (2000) approach is the introduction of four crucial criteria to have a well-defined semantics for ACL messages. In this article, we explore and evaluate how current proposals in this area of research are handling feature dimensions of those criteria. The criteria are as follows:

1. *Formal*: the language must be formal (i.e. there exists a formal syntax and semantics) to eliminate the possibility of ambiguity in the meaning of ACL messages and, in principle, allow agents to reason about them.
2. *Declarative*: the semantics should focus on what social interactions and protocols are about (i.e. focusing on the aims of the interactions) so as to avoid over-constraining the interactions by specifying how interactions should be accomplished. Furthermore, logical languages and reasoning techniques are central to achieve declarative criterion.
3. *Meaningful*: the semantics should focus on the content and meaning of messages, not on their representation as sequences of tokens to enable agents to deal better with exceptions (Kafali & Torroni, 2012) and opportunities.

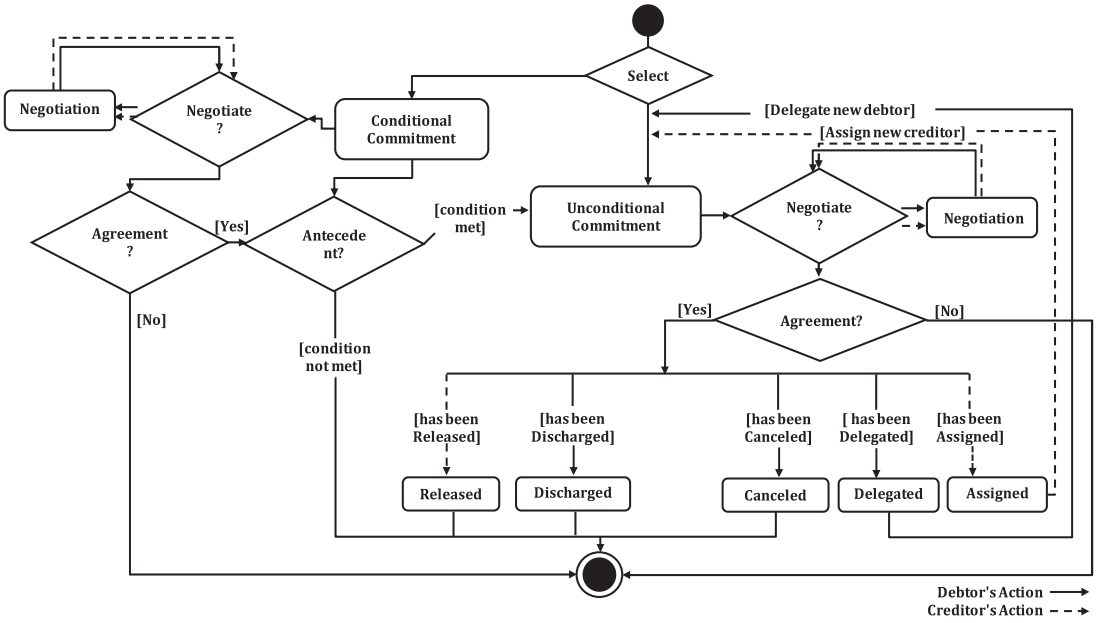

Figure 1 A commitment life cycle by making use of an UML state diagram. The rounded rectangles represent the states and the directed edges represent the transitions. In particular, the label of a rectangle is the commitment state, whereas the label of an edge is an action or an event that causes the transition

4. *Verifiable*: we can check if agents are acting according to the given semantics and/or protocols. The verification tool is evaluated in seventh criterion. Those criteria are agent communication driven. As our focus is on commitments and their verification methods, we consider three additional commitment-oriented criteria.
5. *Commitment modeling*: how commitment is formally modeled as, for example, proposition, predicate, fluent, and temporal modality. By knowing the way of modeling commitments, we can check whether the commitments have a real and concrete semantics or not. If so, we can apply sixth criterion.
6. *Commitment semantics*: we use to check whether or not there exists a formal semantics for commitments. Having formal semantics, we can apply the fourth criterion and prove soundness.
7. *Verification method*: what is the tool exploited to verify the correctness of commitments and associated actions along with commitment-based protocols with specifications (e.g. model checking, theorem proving, and monitoring tool). This criterion can help designers select a verification method based on their own needs. For example, some designers favor model checking to detect design errors before the actual implementation is undertaken.

It is worth noticing that the verifiable criterion checks the possibility of testing the behavior of interacting agents with respect to the given semantics and/or protocols. Having computationally grounded semantics, we can directly define concrete models (or scenarios) that we use in the verification process. This process is always performed at design time. On the other hand, the compliance of agents' behaviors are tested with the protocol specifications (used as concrete scenarios). It can be performed at run time or design time. In both cases, what are the used tools and techniques to perform the verification process? The verification method criterion is responsible to answer this question. Moreover, the real difference between the meaningful criterion and commitment semantics criterion is as follows: (1) the meaningful criterion is agent communication driven aspect as it requires that the proposed semantics should focus on the content and meaning of messages, not on their representation as sequences of tokens; and (2) the commitment semantics is one criterion of commitment-oriented criteria as it only focuses on how commitment is formally captured (as predicate, temporal operator, etc.), that is, it is mainly related to the formal semantics of commitments, not commitment actions. In social commitment-based approaches, the meanings of

messages (or protocol actions) are directly captured by commitment actions, which transform commitment state into another state. These meanings are known and agreed upon by agents before they participate in the protocol. However, agent is free to execute any action as long as it respects the protocol rules.

3 Running example

We choose the NetBill protocol (NB) (Sirbu, 1997) to demonstrate our presentation and discussion of current proposals and to evaluate them. This choice is mainly motivated by the fact that the NB is widely and commonly used in the literature to show how social commitments can specify protocols in business settings. It is a business protocol, developed for buying, purchasing, and selling goods over the Internet. The protocol regulates and coordinates the interaction between two roles, named customer (*Cus*) and merchant (*Mer*). As a matter of fact, we adopt an enhanced version of the protocol, which provides extra choices to the participating agents in order to enable them to react better to emerging opportunities, and to better handle exceptions, which might arise when the interaction starts (Chopra & Singh, 2004; Yolum & Singh, 2004; Winikoff *et al.*, 2005). Figure 2 shows a commitment machine of the protocol where (1) each state is labeled by a proposition representing certain fact as well as a commitment holding at this state; and (2) each transition is labeled by an agent and its action. The figure is self-descriptive. $C1$, $C2$, and $C3$, respectively, represent commitments: C (*Mer*, *Cus*, *deliverGoods*), C (*Cus*, *Mer*, *sendPayment*), and C (*Mer*, *Cus*, *sendReceipt*) stating that *Mer* commits to deliver the request goods to *Cus*, *Cus* commits to send the agreed payment to *Mer*, and *Mer* commits to send the receipt to *Cus*. In order to flexibly specify the protocol, social commitment-based approaches enable us to start the interaction by: (1) *Mer* can present an offer without receiving a request from *Cus* (as it happens for advertising): $C4 = CC$ (*Mer*, *Cus*, $\top$, *presentOffer*) where $\top$ is 'propositional constant true'; (2) *Mer* can commit to deliver some goods for trial without asking *Cus* for accepting the price: $C5 = CC$ (*Mer*, *Cus*, $\top$, *deliverGoods*); or (3) *Cus* can accept the price quote before *Mer* proposes one: $C6 = CC$ (*Cus*, *Mer*, $\top$, *acceptOffer*), mimicking *Cus*'s trust on the fact that the merchant will make an offer.

4 Linear temporal logic and commitment-based agent communication

LTL was first proposed by Pnueli (1977) as a formalism to specify properties of concurrent programs and to reason about their behavior. In LTL perspective, each moment in time has a unique possible future.

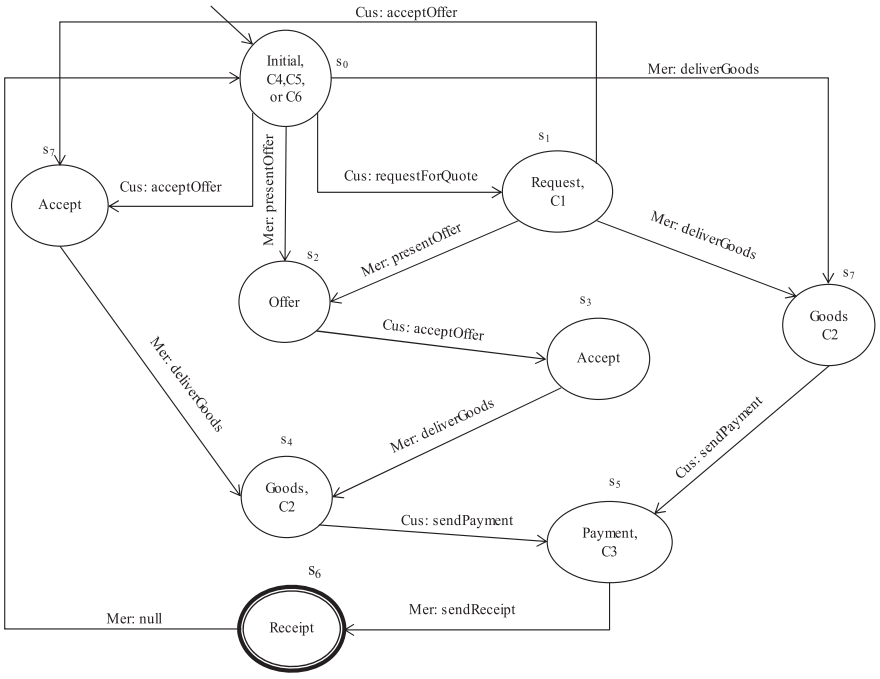

Figure 2 A commitment machine of the enhanced NetBill protocol

Temporal modalities are thus provided for describing events along a single time line. The basic ingredients of LTL formulae are built up from atomic propositions, Boolean connectives ($\wedge$, $\vee$, $\neg$, $\supset$, etc.) and two basic temporal modalities $\Diamond\varphi$ ('sometime' φ; also read as 'eventually' φ), $\Box\varphi$ ('always' φ; also read as 'globally' φ), $\bigcirc\varphi$ ('next time' φ), and $\varphi\, U\, \psi$ (φ 'until' ψ).

DEFINITION 1 *Given a set $\mathcal{PV}$ of atomic propositions, the syntax of LTL is given by BNF grammar as follows (Clarke et al., 1999):*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc\varphi \mid \varphi\, U\, \varphi$$

where $p \in \mathcal{PV}$ is an atomic proposition, the formula $\bigcirc\varphi$ holds at the current moment, if φ holds in the next moment, and the formula $\varphi\, U\, \psi$ holds at the current moment, if there is some future moment for which ψ holds and φ holds at all moments until that future moment. Other formulae can be introduced as abbreviations in the usual way. In particular, $\varphi \supset \psi$ abbreviates $\neg\varphi \vee \psi$, $\Diamond\varphi$ abbreviates $(\top\, U\, \varphi)$, and $\Box\varphi$ abbreviates $\neg\Diamond\neg\varphi$. Intuitively, $\Diamond\varphi$ stands for φ will be true in the future moment.

The semantics of LTL formulae is given in terms of *transition systems*. A transition system $M = (S, R_t, V, I)$ is a tuple where S is a non-empty set of states, $R_t \subseteq S \times S$ is a transition relation, $V : S \to 2^{\mathcal{PV}}$ is an evaluation function, and $I \subseteq S$ is a set of initial states. The transition relation R_t models temporal transitions among states: given two states $s, s' \in S$, $(s, s') \in R_t$ means that s' is an immediate successor of s. Conventionally, it is assumed that every state has a successor state, that is the transition relation is total. Because most concurrent systems are designed not to halt during normal execution, one can model a computation *path* π in M as an infinite sequence $\pi = (s_0, s_1, \ldots)$ of states such that $(s_i, s_{i+1}) \in R_t$ for all $i \geq 0$. $\pi(i)$ is the ith state in the path π. π_i is the part of π starting at $\pi\,(i)$, that is, $\pi_i = \pi(i), \pi(i+1), \ldots$

DEFINITION 2 *Satisfaction of an LTL formula φ with respect to a path π in a transition system M, denoted by $M, \pi \models \varphi$, is inductively defined as follows:*

- $M, \pi \models p$ *iff* $p \in V(\pi(0))$
- $M, \pi \models \neg\varphi$ *iff* $M, \pi \not\models \varphi$
- $M, \pi \models \varphi \vee \psi$ *iff* $M, \pi \models \varphi$ *or* $\pi \models \psi$
- $M, \pi \models \bigcirc\varphi$ *iff* $M, \pi_1 \models \varphi$,
- $M, \pi \models \varphi\, U\, \psi$ *iff* $\exists k \geq 0$ *such that* $M, \pi_k \models \psi$ *and* $M, \pi_i \models \varphi\, \forall\, 0 \leq i < k$
- $M, \pi \models \Box\varphi$ *iff* $M, \pi_k \models \varphi\, \forall\, k \geq 0$

According to this satisfaction relation, an LTL formula φ holds at s iff all paths starting at s satisfy φ, it is written as $s \models \varphi$.

In the rest of this section, LTL modalities are extended with propositional dynamic logic (DL), new operators such as linear implication and linear operators for capabilities and resources, or past modality. The resulting logical language caters a natural mechanism to (1) specify commitment-based protocols; (2) define protocol actions and their effects; and (3) express the consequence of commitments. LTL modalities are also enriched with modalities to represent and reason about commitments. Furthermore, LTL is exploited to express business properties and formalize regulative protocol specifications.

4.1 Giordano and colleagues

Giordano *et al.* (2003, 2007) developed a logical framework based on *Dynamic Linear Time Temporal Logic* (DLTL), an extension of LTL with *propositional dynamic logic*, to specifically specify and verify commitment-based protocols. In this framework, the protocol describes the meaning of communicative actions in terms of their effects on the system's social state, including commitments and some facts related to the protocol's execution. These effects are expressed by means of 'action laws'. The protocol specifies a set of 'precondition laws', which, in principle, define the execution of actions, a set of 'causal laws' capturing the dependencies among social states, and a set of temporal constraints, which provide restrictions on the possible correct behaviors of the protocol.

Example 4 *An example of an action law from Mer's perspective is that whenever Mer delivers the goods, then it will commit to send the receipt if Cus sends the agreed payment:* $\Box([deliverGoods] \supset (Goods \wedge CC(Mer, Cus, sendPayment, sendReceipt)))$.

Example 5 *The precondition law for Mer is as follows:* $\Box(\neg Payment \supset [sendReceipt] \perp)$ *meaning that whenever the payment has not been made by Cus, the action sendReceipt cannot be executed (the notation* $[a] \perp$ *means there is no resulting state following the execution of the action a).*

In this approach, commitments are modeled as fluents, basic components of EC (Shanahan, 2000). In principle, every fluent is either true (hence commitment holds) or false (hence commitment does not hold), and no fluent is both true and false. A fluent might also exchange its truth value with the performance of communicative actions. For instance, the fluent *C*(*NB*, *Mer*, *Cus*, *deliverGoods*) means that *Mer* is committed to *Cus* to execute the action *deliverGoods* in *NB* (cf. Figure 2). However, the approach does not explicitly introduce commitment actions. Instead, the authors introduced a causal law to define the operational semantics of discharge action.

Example 6 *The causal law for discharging the commitment C (NB, Mer, Cus, deliverGoods) is delineated by indicating that in the next state (s), the fluent representing the commitment becomes false whenever deliverGoods holds in that state (i.e. s). This would be expressed as follows (Giordano et al., 2007):* $\Box(\bigcirc deliverGoods \supset \bigcirc \neg C(NB, Mer, Cus, deliverGoods))$.

It is worth mentioning that Singh (2008) removed the temporal operators $\Box$ and $\bigcirc$ in the postulate that defines the operational semantics of discharging commitments in order to make it reasonable and intuitive. In the verification part, the authors discussed different types of verification problems depending on whether the verification process is carried out at design time such as verifying protocol properties or at run time such as verifying agents compliance with a protocol. However, they informally reduced DLTL formulae into standard LTL formulae, and commitment-based protocol into PROMELA (the input language of the SPIN model checker (Holzmann, 1997)) wherein commitments are represented as PROMELA processes and protocol actions are represented as PROMELA message channels, which are communicating interleaved processes. By informally, we mean that the reduction (transformation) rules are not delineated by a systematic and formal way; so the soundness of the mapping from DLTL formulae into LTL formulae can be proved. The semantics of this approach is formal, declarative, meaningful, and verifiable using model checking. Model checking is a full, formal, and automatic verification technique during the design time. By modeling commitments as fluents, we will have the issue discussed in the following remark.

Remark 1 *While modeling commitments as fluents, which are treated as atomic propositions, is easy to implement and treat, it suffers in our perspective from three technical shortcomings: (1) it does not capture the computationally grounded meanings of commitments, which can be interpreted directly into concrete models as it focuses only on the reference (i.e. truth values); (2) the social and directive communication aspects between two interacting agents in the context of commitment are not captured as in fluent modeling, we do not know which agent is the debtor and which agent is the creditor; and (3) no formal semantics is given to commitments.*

In fact, the problem in Remark 1 is not about the expressiveness of fluent representation, but it is in the fluent modeling itself. Specifically, by modeling commitment $C(i, j, \varphi)$ as a fluent, the truth value of this fluent after substituting i and j and then evaluating the truth value of φ is either true (i.e. commitment holds) or false (i.e. commitment does not hold) (the solution of this problem is introduced after Remark 3). Furthermore, the use of the SPIN model checker has a limitation discussed in Remark 2.

Remark 2 *Because of the state explosion problem*[6] *of automata-based model checking techniques (Clarke & Emerson, 1982; Clarke et al., 1999), the SPIN model checker is usually not applicable in open systems (e.g. protocols) having a large state space.*

[6] The state explosion problem means that the number of global states in a model grows exponentially with the number of variables, or concurrent components, constituting the modeled system.

4.2 Pham and colleagues

Pham and Harland (2007) introduced an approach that extends LTL with linear operators that model resources, actions, and capabilities of interacting agents, the resulting logical language is named TLL. This language is in turn exploited to flexibly generate commitment-based protocols from the internal view of participating agents in terms of a set of agents' capabilities, resources, and pre-commitments. Those pre-commitments can be negotiated among each other using inference rules and upon being accepted, will form conditional commitments. The authors exploited negative formulae, which can be regarded as formulae in demand and linear implication expressing the causal relationship in the form of reasoning rules to represent and reason about unconditional and conditional commitments, respectively. When the antecedents are satisfied, the linear implication will assure that unconditional commitments turn effective. An unconditional commitment is fulfilled whenever the agent successfully performs the actions required by its commitment. This specifically means that the corresponding positive formula—which can be regarded as formula in supply—in TLL is derived.

Example 7 *Let a good item* (*resource*) *'g' located at* (*denoted by '@'*) *and owned by Mer be denoted by* $item_g@Mer$ *and a commitment of Mer to deliver that item be represented in a negative formula—which can be viewed as formula in demand—as* $item_g@Mer^{\perp}$. *Then, the positive formula* $item_g@Mer$ *will fulfill the commitment* $item_g@Mer^{\perp}$ *and the negative and positive formulae are automatically removed* (*i.e. the commitment in demand is resolved*): $item_a@Mer \otimes item_a@Mer^{\perp} \vdash \perp$, *where* $\otimes$ *is a linear multiplicative conjunction and* $\vdash$ *is an inference relation.*

Example 8 *A commitment* $item_a@Mer^{\perp}$ *can be canceled, denoted by* $(Cond \otimes item_a@Mer)^{\perp}$, *where the token Cond is simply generated by the agent's internal database when the agent breaks its commitment* $item_a@Mer$. *As for the case of fulfilling commitment, once Cond is generated, due to the inference rule* $Cond \otimes (Cond \otimes item_a@Mer)^{\perp} \vdash \perp$, *the commitment* $item_a@Mer^{\perp}$ *is removed and henceforth canceling the commitment of deriving* $item_a@Mer$.

However, the way of generating the token *Cond* is not considered. Constructing protocols from the view of participating agents by making use of proof search and putting the specification of commitments locally at the respective agents according to their roles has an interesting advantage toward avoiding the mapping 'from protocol actions to commitment operations'. However, some issues are missing in this approach. The first one is syntactic: the realm of commitment has only the debtor, which is not an effective representation as the two main features of commitment representation are social and directive. The second one is in modeling commitments as propositions, which in turn waive formal semantics (cf. Remark 1). The third one is of adopting the linear implication to obtain an unconditional commitment from the conditional one when its antecedent is true. The linear implication '$\multimap$' represents capabilities of agents in producing and consuming resources. Let p and q be agents' resources, so $p \multimap q$ ensures that the antecedent before p will be transformed into the antecedent after q in which p is removed after producing q. We can think of the linear implication as the one Canadian dollar p is 'gone after using it to buy a chocolate bar' q. From our perspective, a linear implication might function as the *material implication* ($p \supset q$), which is true when p is false, so we could have an unconditional commitment without satisfying its antecedent. To evaluate the semantics of this approach, it meets the formal (in terms of introducing logic-based approach to define the meaning of protocol actions via commitments) and meaningful criteria, but it does not consider the verifiability issue. However, as protocols are specified in terms of what is to be achieved rather than how the agents should act, the proposed semantics meets the declarative criterion.

4.3 Singh and colleagues

Desai *et al*. (2007) proposed an interesting way to model business protocols in terms of social commitments and their actions, which in turn reduce the emphasis on the rigid sequence of actions that participating agents must take and facilitate loose coupling among these agents. They also exploited a particular language named OWL-P introduced in Desai *et al.* (2005) to model business protocols. OWL-P supports the specification and composition constructs to build standard business protocols and incorporates the

standard web ontology language (OWL) for modeling well-defined business processes. The authors argued that the correct composition of business processes can be expressed via individual protocols and their composition constraints and thereby enabling the verification of a wide range of composed processes. To verify properties geared toward the composition of protocols, the authors informally reduced protocols into the PROMELA language, which in principle allows them to model the composition of protocols in terms of a set of processes and analyzing the resulting model with the SPIN model checker. In this technique, as in Giordano *et al.* (2007), commitments and their actions are modeled in terms of OWL-P rules, which are mapped, respectively, into PROMELA processes without capturing all features of commitments such as evaluating the consequence of commitment at the proper time, and into PROMELA messages. The checked properties are classified into general properties such as *deadlock* and *livelock* freedom and *protocol-specific properties* that check a specific behavior of participating agents. An example of protocol-specific property is as follows.

Example 9 $\Box(Payment \supset \Diamond(C\ (Mer, Cus, sendReceipt)))$*, meaning that when Cus sends the payment, Mer should eventually commit to send the receipt to Cus* (*cf. Figure 2*).

It is clear that this semantics is formal, declarative, meaningful, and verifiable. Although the authors focused on modeling commitments as simple processes in the PROMELA language to address the lack of communication aspect raised in fluent modeling, no formal semantics is given to commitments. Recall that a formal LTL-based framework is in fact proposed to solely express protocol properties. To this end, three missing points in this approach are as follows: (1) a formal model for protocols; (2) a formal translation procedure; and (3) a formal semantics for commitment actions themselves.

Singh (2008) introduced the model-theoretic semantics for social commitments by postulating some rules (axioms) that can be utilized as methods to reason about commitments and solely detach and discharge actions. A commitment becomes detached when its antecedent is true. A commitment that is detached but fails to be discharged indicates a violation. This model particularly extends LTL with modalities for two kinds of commitments: *practical commitments*, which are about what is to be made, whereas *dialectical commitments*, which are about what holds. In this model, the semantics is interpreted using Segerberg's idea, which in turn maps 'each world into a set of set of worlds'. For instance, to define the semantics of dialectical commitments, Singh introduced a function:

$$\mathbb{D} : \mathbb{T} \times \mathcal{A} \times \mathcal{A} \times \mathscr{P}(\mathbb{T}) \rightarrow \mathscr{P}(\mathscr{P}(\mathbb{T}))$$

that produces a set of propositions for each moment, an ordered pair of two agents and proposition where $\mathbb{T}$ is a set of moments, $\mathcal{A}$ is a set of agents, and $\mathscr{P}(\mathbb{T})$ is the powerset of $\mathbb{T}$. Each proposition is a set of moments. This function yields a set where each member is a consequent proposition, which in turn captures what the debtor would be committed to if the antecedent is met. Thus, the commitment semantics is delineated by computing the set of moments where the consequence holds $[\![\varphi]\!]$ and testing if those moments are among the moments $[\![\tau]\!]$ computed by $\mathbb{D}$ on the commitment antecedent τ:

$$M, s \models CC\ (i, k, \tau, \varphi)\ \textit{iff}\ [\![\varphi]\!] \in \mathbb{D}(s, [\![\tau]\!])$$

Notably, the author focused on identifying the logical terms between the antecedent and consequence of commitment to avoid the problem of both the linear implication exploited in Pham and Harland (2007) and the 'strict implication' advocated in Singh (2000) to define the semantics of conditional commitments (cf. Section 5 for the problem of strict implication).

Example 10 *From Figure 2, when the proposition Payment holds* (*i.e. Cus sends successfully the payment to Mer*)*, the commitment with the consequent sendReceipt comes into being, and when the proposition Receipt is true* (*i.e. Mer has successfully sent the receipt to Cus*)*, the commitment is discharged and no longer active:*

- $CC\ (Mer, Cus, sendPayment, sendReceipt) \wedge Payment \supset C\ (Mer, Cus, sendReceipt)$;
- $Receipt \supset \neg CC\ (Mer, Cus, sendPayment, sendReceipt)$.

It is obvious that this semantics is formal, declarative, and meaningful. However, the computation of the function $\mathbb{D}$ is not specified, which feasibly makes the possibility of implementing and applying such an

approach unclear as discussed in Bentahar *et al.*(2012). As the semantics is not based on Kripke structures, verifying such a semantics using model checking needs either defining an equivalent semantics using Kripke structures or interpreted systems, or defining a completely new model checking approach for the extended logic from scratch. The semantics of discharge action focuses upon checking whether the truth condition of the commitment consequence is true or not, but we believe this should be a part of the semantics of the commitment modality not the discharge one. Throughout the article, we refer to this problem as *over-specification*. Simply put, such a problem occurs when some specifications are repeated in the semantics of different operators. In our case, this problem happens because the truth condition of the commitment consequence is part of not only the commitment operator, but also its discharge.

4.4 Baldoni and colleagues

Baldoni *et al.* (2010, 2011, 2013) observed that current specifications of commitment-based protocols do not account for the *regulative* specifications and focus only on the *constitutive* specifications. The constitutive specification defines the semantics of all agents' actions in terms of how these actions affect the social state, whereas the regulative specification constraints the evolution of the social state. They forcefully argued that the two specifications together define the meaning of the interaction (because the only constraint by which we can say that an interaction governed by protocol is successful is that all commitments are discharged), but it is well suited to distinguish between them. The authors then introduced a framework to model commitment-based protocols that separates the constitutive and regulative specifications. They also showed that the regulative specifications should be based on commitments holding in social states not on the execution of actions in order to address the limitations raised in previous proposals (Giordano *et al.*, 2003, 2007) regarding the *openness, interoperability*, and *modularity* of designing MASs, which in fact complicate the re-use of software agents' actions. Furthermore, the authors defined a *first-class* and *declarative* language, named, `2CL`, to represent the constraints among commitments (i.e. only the regulative specifications). `2CL` provides seven kinds of constraint rules, such as *correlation*, *co-existence*, *cause*, *response*, and *before*. Such constraint rules are delineated by making use of equivalent LTL temporal modalities.

Example 11 *An example of the before constraint rule is as: Payment* $\rightarrow\!\triangleright$ *C* (*Mer*, *Cus*, *sendReceipt*) *means that Mer cannot commit to send the receipt to Cus before* ($\rightarrow\!\triangleright$) *Cus has sent the agreed payment.*

Example 12 *In LTL, the before constraint rule can be expressed as:* $\neg(\neg$*Payment U C* (*Mer*, *Cus*, *sendReceipt*)).

Notice that the commitment of the merchant is modeled as an LTL atomic proposition.

Example 13 *The cause constraint rule is given as: Payment* $\rightarrow\!\triangleright$ *Receipt meaning that if Cus pays, then Mer is expected to send the receipt and the receipt should be only sent after the payment.*

The proposed semantics is declarative and meaningful. When the protocol specification is exploited to define the meaning of ACL messages, the semantics of this approach satisfies the formal criterion. However, the approach waives the formal semantics for commitments as commitments are simply modeled by propositions, which in principle suffer from the same problem discussed in Remark 1. Although the constraint rules can be expressed directly by LTL modalities, which in turn opens the way to utilize different model checkers to check, for example, whether or not agent behaviors comply with the protocol, the authors do not consider the verification issues. To this end, separating the regulative rules from commitments has been criticized recently by Marengo *et al.* (2011). In particular, Marengo *et al.* placed regulative rules in the antecedents and consequences of commitments to identify the 'duties and rights' of the debtor and creditor.

4.5 Verdicchio and colleagues

Spoletini and Verdicchio (2007) developed an 'automata-based monitoring module' to keep the interactions of a single agent safe by checking whether the evolution of its commitment states with regard to

certain events is compatible with some desirable properties defined in agent's specifications and expressed in LTL$^{\pm}$ (an extension of LTL with past and future modalities). A commitment is modeled as a predicate with four components: an event e that has created the commitment, a debtor i, a creditor j, and a consequence φ: $Comm\ (e, i, j, \varphi)$. The consequence φ of a commitment is represented by a first-order term and written $\lfloor\varphi\rfloor$ in order to refer to the relevant LTL$^{\pm}$ formula in which $\lfloor\varphi\rfloor$ is named a 'truth-preserving translation' of φ. The commitment is fulfilled (resp. violated) when its consequence is true (resp. false):

$$Fulf\ (e, i, j, \varphi) \triangleq Comm\ (e, i, j, \varphi) \wedge \lfloor\varphi\rfloor$$
$$Viol\ (e, i, j, \varphi) \triangleq Comm\ (e, i, j, \varphi) \wedge \neg\lfloor\varphi\rfloor$$

In the verification part, the monitoring module comprises of two components (Spoletini & Verdicchio, 2009): (1) 'Word Composer' responsible for collecting all data resulting from agent communication, preserving track of the relevant information, and in turn elaborating them to assess and evaluate present and past time-stamped words selected from these data; and (2) 'Word Analyzer' that translates such words into Alternating Modulo Counting Automata (Spoletini, 2005) functioning as a 'language acceptor'. Unaccepted word in this approach means that the state of the system does not satisfy the property expressed by the monitored formula, and such violation is notified to the agent. This semantics meets formal (using LTL$^{\pm}$-based framework to define the semantics for agent communication), declarative (wherein every communicative action can be viewed as the modification of commitments attaching agents to others), meaningful, and verifiable criteria. However, this approach lacks formal semantics for commitments because commitments are modeled as predicates, which are reduced into propositions as stated by the authors as 'to be able to write the properties to be monitored in the form of LTL$^{\pm}$ formulae, we need to translate the first-order logic sentences into propositions'. Furthermore, the semantics of fulfillment and violation actions suffer from the over-specification problem. It is worth noticing that whereas modeling commitments as predicates by making use of a pure predicate logic (or first-order logic) (Huth & Ryan, 2004) allows one to model real-life business scenarios, not only they cannot distinguish among various 'modes' of truth, such as inevitably true, $\Box$, and true in the future, $\Diamond$, but also suffer from the problem reported in the following remark.

Remark 3 *Consider the following example, a customer Cus commits to pay* \$100 *to a merchant Mer. A naive attempt to translate this declarative sentence into the pure predicate logic might result in the following: C*(*Cus*, *Mer*, *pay* \$100)*. However, the commitment notion is* `referentially opaque`*—*which set up `opaque contexts` *in which the standard substitution rules usually exploited in the pure predicate logic* (*Huth & Ryan, 2004*) *cannot be applied. To illustrate this issue, let us assume that* \$100 *is equivalent to* €81 (*for a particular day*)*, but we cannot simply say that the customer commits to the merchant to pay* €81 (*i.e. C*(*Cus*, *Mer*, *pay* €81)) *is equivalent to the customer commits to the merchant to pay* \$100 (*i.e. C*(*Cus, Mer, pay* \$100)) *because the commitments C*(*Cus, Mer, pay* \$100) *and C*(*Cus, Mer, pay* €81) *are considered in the pure predicate logic as different commitments although they are equivalent in all the worlds where* \$100 = €81*. Recall that semantic value of each term in the pure predicate logic is dependent only on the denotations of its arguments. The operators of predicate logic are said to be truth functional. On the other hand, commitments are not truth functional. Thus, what is meant by referentially opacity is 'substituting equivalents into opaque context is not going to preserve meaning' as Wooldridge has eloquently argued in Wooldridge* (*2009*).

Indeed, the problem in Remark 3 is not about the flexibility of protocol (e.g. to say that the protocol should consist of several ways of the payment), but it is about the possibility of considering equivalent commitments as different commitments. To solve the problems in Remarks 1 and 3, the 'sense' of the formula having opaque context is needed along with its truth value. A similar solution was introduced by Wooldridge (2009), who models agent's belief as a temporal modality. In principle, the sense of commitment $C\ (i, j, \varphi)$ is to qualify the truth value of φ, that is $C\ (i, j, \varphi)$ is satisfied at a state s iff φ is not only true in a certain state but true in all accessible states s' from s defined by certain accessibility relation. This accessibility relation is defined by considering the local states of interacting agents as well as other conditions to generate a computationally grounded semantics, which is entirely missing in fluent

(or predicate) modeling and this is why EC-based approaches do not have formal models. The main advantage of computationally grounded semantics is to directly interpret formulae into concrete models and vice versa.

In the literature of agent communication semantics, there are few proposals that have been introduced to address the limitation of the pure predicate logic and the lack of communication aspect emerged from fluent and proposition modelings. For instance, Colombetti (2000) proposed an extended first-order modal language (EFOM) to represent and reason about commitments. In particular, the author introduced a new set of basic speech-act-based ACL, which is named Albatross (agent language based on a treatment of social semantics). This ACL is based on the social notion of commitments. The semantics of the main elements of Albatross is based on the EFOM language. Concretely, the semantics of commitment *modality* is delineated by using a certain type of accessibility relation: $f_c : D_{\text{action}} \times D_{\text{agent}} \times D_{\text{agent}} \times S \rightarrow 2^{2^S}$, which in turn produces a family set of states for each state, and a typed domain of individual sorts of action and a pair of two agents. The semantics is defined by computing the set $\|\varphi\|$ of states satisfying the commitment consequence and testing if this set is among the set of states computed by f_c. This would be formally defined as follows:

$$M, s \models C(e, i, j, \varphi) \textit{ iff } \| \varphi \| \in f_c(\delta(e, s), \delta(i, s), \delta(j, s), s)$$

where $C(e, i, j, \varphi)$ is named a *modal predicate* and means that the performance of action e commits i relative to j to bring about φ, $\delta(e, s)$ and $\delta(x, s)$, where $x \in \{i, j\}$ define, respectively, the denotation of action e and agent x at s. The author showed that the commitment is violated when its consequence is false. A violation of the commitment $C(e, i, j, \varphi)$ is delineated by the following semantic rule:

$$V(e, i, j) \triangleq C(e, i, j, \varphi) \wedge \neg\varphi$$

As claimed by the author, the logic of commitment is 'indeed very weak' and there is no 'counterpart of the *D* axiom'. Consequently, an agent can make conflicting commitments. The author also contended that inferential capacities are 'sufficient to derive that at least one of two conflicting commitments is going to be violated'. He in turn showed that reasoning upon violations provides a way to deal with conditional commitments in question. This semantics is formal, declarative, and meaningful, but verifiability has not been addressed. As in Singh (2008), the approach does not describe how in practice the accessibility relation f_c can be computed. The approach also does not define (1) axioms of commitment logic; and (2) the semantics of other commitment actions to automatically reason about the developed language.

We conclude this section with Table 1, which in principle summarizes the results of evaluating the contributed proposals that advocate LTL (or an extension of LTL) to define the semantics for ACL messages. In the table, we use **For.**, **Dec.**, **Mea.**, and **Ver.**, to, respectively, refer to formal, declarative, meaningful, and verifiable Singh's criteria. We also indicate how the commitments are modeled (**Mod.**) and if a formal semantics (**Sem.**) is defined for them. Although all reported proposals satisfy the first three criteria (formal, declarative, and meaningful), there is no formal semantics for social commitments, except Singh's (2008) proposal. The declarative specification of commitment-based protocols will permit agents to realize in advance whether or not the protocol supplies them enough assures of attaining their objectives to take part.

We claim that our new criteria would help designers and developers decide which proposal will satisfy their choices or will meet their needs. To support this claim, we consider Desai *et al.*'s (2007) approach and Spoletini and Verdicchio's (2007, 2009) proposals as illustrative examples. Although Desai *et al.*'s approach and Spoletini and Verdicchio's proposals satisfy all agent communication semantics criteria, their evaluations with respect to commitment-oriented criteria and verification method criterion are entirely different. If designers need to verify the correctness of the developed semantics before proceeding to implement it in order to detect design errors, they will select Desai *et al.*'s proposal, which adopts a model checking technique, applied at design time, and they know the semantics of commitments will be transformed into PROMELA processes to be able to use SPIN. On the other hand, if designers

Table 1 Summary of linear temporal logic and commitment-based agent communication

	Agent communication semantics				Commitment		
	For.	Dec.	Mea.	Ver.	Mod.	Sem.	Verification method
Giordano *et al*. (2003, 2007)	√	√	√	√	Fluent		Model checking (informal translation to SPIN)
Pham and Harland (2007)	√	√	√		Formula in demand		
Desai *et al*. (2007)	√	√	√	√	PROMELA process		Model checking (informal translation to SPIN)
Singh (2008)	√	√	√		Temporal modality	√	
Baldoni *et al*. (2010, 2011)	√	√	√		Proposition		
Spoletini and Verdicchio (2007, 2009)	√	√	√	√	Predicate		Monitoring tool

plan to monitor the behaviors of interacting agents at run time, they will use monitoring tool developed by Spoletini *et al.* that represent commitments as predicates. Moreover, as there are no experimental results reported in all reviewed proposals in Table 1, we could not discuss the performance of the used verification methods. When we classify a particular semantics of a proposal as being not verifiable, we mean that the verifiability issue is not considered, but still there might be a possibility to use a technique to verify that proposal.

5 Computation tree logic and commitment-based agent communication

In early 1980s, another strand of temporal logic named CTL for specification and verification purposes was introduced in Clarke and Emerson (1982). The underlying structure of time in CTL is assumed to have a branching tree-like nature, which refers to the fact that each moment in time might be split into several possible futures, which enables CTL to capture the non-determinism. CTL requires that a path quantifier—either A ('for all paths') or E ('for some path')—is immediately followed by a single operator from the list of usual temporal operators: $\Box$ ('globally'), $\Diamond$ ('sometime'), $\bigcirc$ ('next time'), and U ('until') to obtain a well-formed state formula.

Definition 3 *The syntax of CTL formulae over the underlying set $\mathcal{PV}$ of atomic propositions is given by a BNF grammar as follows (Clarke et al., 1999):*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid E \bigcirc \varphi \mid E\Box\varphi \mid E(\varphi\, U\, \varphi)$$

where $p \in \mathcal{PV}$ is an atomic proposition, $E \bigcirc \varphi$ is read 'there exists a path such that at the next state φ holds', $E\Box\varphi$ is read 'there exists a path such that φ holds globally along the path', and $E(\varphi\, U\, \psi)$ is read 'there exists a path such that φ holds until ψ holds'.

The syntax of CTL includes the following temporal operators as abbreviations: $A \bigcirc \varphi$, $A\Box\varphi$, $A(\varphi\, U\, \psi)$, $E\Diamond\varphi$, and $A\Diamond\varphi$. Notice that the universal path quantifier A can be delineated by the existential path quantifier and negation, for example: $A \bigcirc \varphi \triangleq \neg E \bigcirc \neg\varphi$. These abbreviations are read, respectively, as: 'for all paths, in the next state φ holds', 'for all paths, φ holds globally', 'for all paths, φ holds until ψ holds', 'there exists a path such that φ holds at some future point', and 'for all paths, φ holds at some point in the future'.

As in LTL, the semantics of CTL formulae is given in terms of a transition system $M = (S, R_t, V, I)$, where S, $R_t \subseteq S \times S$, $V : S \to 2^{\mathcal{PV}}$ and $I \subseteq S$ have the same meaning as in LTL. In addition, a computation path π in M is an infinite sequence $\pi = (s_0, s_1, \ldots)$ of states such that $(s_i, s_{i+1}) \in R_t$ for all $i \geq 0$. $\pi(i)$ is the ith state in the path π.

DEFINITION 4 *We use the standard notation to indicate truth in a transition system: M, s* $\models \varphi$ *means that formula* φ *holds at state s in a transition system M. The relation* $\models$ *is defined inductively as:*

- $M, s \models p$ *iff* $p \in V(s)$
- $M, s \models \neg\varphi$ *iff* $M, s \not\models \varphi$
- $M, s \models \varphi \vee \psi$ *iff* $M, s \models \varphi$ *or* $M, s \models \psi$
- $M, s \models E \bigcirc \varphi$ *iff* $\exists \pi$ *such that* $\pi(0) = s$ *and* $M, \pi(1) \models \varphi$
- $M, s \models E\Box\varphi$ *iff* $\exists \pi$ *such that* $\pi(0) = s$ *and* $M, \pi(i) \models \varphi \forall i \geq 0$
- $M, s \models E(\varphi U \psi)$ *iff* $\exists \pi$ *and for some* $k \geq 0$ *such that* $\pi(0) = s$ *we have* $M, \pi(k) \models \psi$ *and* $M, \pi(i) \models \varphi \forall 0 \leq i < k$

The interpretations of atomic propositions, Boolean operators, and temporal modalities are as usual (see, e.g. Clarke *et al.*, 1999).

In the rest of this section, we investigate the role of CTL enriched with modalities or predicates to represent and reason about commitments and associated actions. The resulting logical language can be then exploited to define the semantics of ACL messages, specify commitment patterns, define a richer temporal aspect of commitment consequences, and specify commitment-based protocols. Such protocols are essential to the functioning of open systems, such as those that arise in most interesting Web applications, service engagements, and business processes.

5.1 Singh and colleagues

Venkatraman and Singh (1999) introduced an approach for locally testing whether the behavior of an agent in Web-based MASs complies with a commitment-based protocol specified in CTL with the idea of 'potential causality'. Their verification process centers on the conditions under which an individual agent (named an observer, who participates in the protocol) can check the satisfaction or violation of commitments made by each agent toward each other by making use of a model checking-like technique. This technique is similar to model checking, but it uses a different procedure and performs at run time. Technically, a model checking-like approach can 'only falsify (but not verify) the correctness of the construction of the agents' in the MAS. That is, when the observer agent discovers an inappropriate execution, this entails that the system does not satisfy the protocol. Each commitment in this approach is a meta-commitment. The consequence of commitment is expressed as a CTL formula and commitment itself is modeled as a variable c of type T, which is an abstract data type (ADT). Operations that can be performed on commitments are modeled as methods having c as an argument.

EXAMPLE 14 *Let* $c = C$ (*Mer*, *Cus*, *deliverGoods*) *be a variable that models a commitment from Mer to Cus to deliver the requested goods. The discharge operation Discharge* (*Mer*, *c*) *fulfills the commitment c by Mer when the proposition representing deliverGoods is true.*

This ADT can be implemented as a module where the module's interface declares the methods corresponding to the type's operations. In this modeling, there is no formal semantics for commitments. However, when it comes to agent communication, the semantics of this approach is formal (as the protocol is specified in CTL, the formal semantics of ACL messages is well defined), declarative, meaningful (as the meaning of every message is expressed in terms of commitments), and verifiable. Their verification technique is an effective one provided that the system under consideration has a small state space. The semantics of commitment actions, on the other hand, is not considered in this work.

Based on Jürgen Habermas's theory of communicative acts (Habermas, 1984), Singh (2000) proposed three-level meanings for ACLs such that each communicative act is associated with three validity claims: (1) *objective claim* meaning that the debtor is committed to send something that is true; (2) *subjective claim* stating that the debtor is committed to sincerity, that is the debtor believes or intends what is communicated; and (3) *practical claim* regarding to the debtor's claim that it is justified in making the communication. Recall that defining the meaning of communicative acts is not different from the meaning of the messages specified in the protocols as argued in the approach proposed in Venkatraman and Singh (1999). To avoid the shortcoming of Venkatraman and Singh (1999) regarding the representation of

commitments as simple variables, the author extended CTL modalities with three modalities for commitments, beliefs, and intentions to model interactions among agents. He particularly presented three accessibility relations to define the semantics of those modalities. For instance, the semantics of the commitment modality is satisfied at s in a model M iff the consequence is true along every accessible path π delineated by using an accessible relation $\mathbb{C}(i, j, G, s)$ and emanating from the commitment state s, this would be expressed as:

$$M, s \models C(i, j, G, p) \textit{ iff } (\forall \pi : \pi \in \mathbb{C}(i, j, G, s) \supset M, \pi(s) \models p)$$

where the accessibility relation $\mathbb{C} : \mathcal{A} \times \mathcal{A} \times \mathcal{A} \times S \rightarrow 2^{\Pi}$, where $\mathcal{A}$ is a set of agents and Π is a set of paths, produces the set of accessible paths along which the commitments made by the debtor i toward the creditor j in the social context G hold at a state $s \in S$. The author exploited the defined modalities to give the formal semantics of the ACL messages.

Example 15 *An informative message inform* (*Mer*, *Cus*, *sendReceipt*) *can be defined as an action creating a commitment C*(*Mer*, *Cus*, *sendReceipt*) *with its sender Mer as debtor, its receiver Cus as creditor, and its consequent asserting the truth of the proposition sendReceipt specified in the consequence of the message* (*objective semantics*).

Example 16 *The subjective and practical semantics of inform*(*Mer*, *Cus*, *sendReceipt*) *are defined as C* (*Mer*, *Cus*, *MerBsendReceipt*) *and C* (*Mer*, *NB*, *inform*(*Mer*, *Cus*, *sendReceipt*) $\rightsquigarrow$ *sendReceipt) where MerBsendReceipt is read as Mer beliefs sendReceipt and* $\rightsquigarrow$ *is the 'strict implication', which requires sendReceipt to hold when inform*(*Mer*, *Cus*, *sendReceipt*) *holds.*

The strict implication is indeed introduced to define the semantics of conditional commitment. Its semantics is given as follows:

$$M, s \models p \rightsquigarrow q \textit{ iff } M, s \models p \textit{ and } (\forall s' : M, s' \models p \supset (\forall s'' : s' \approx s'' \supset M, s'' \models q))$$

The formula $p \rightsquigarrow q$ is satisfied at s in a model M iff p holds in the current state s and for all states s′ in M, if p holds, then we have that q holds in all states s'' similar to s', that is $s' \approx s''$. Given a commitment-based semantics for ACL primitives, the author exploited it to derive a specification language of commitment-based protocols. The new specification helps analyze the protocol by determining, for example, the compliance of an agent with respect to a given protocol.

Although Singh's approach satisfies formal, declarative and meaningful criteria, it does not clarify the intuition that the accessibility relation $\mathbb{C}$ captures and how accessible paths are computed. More precisely, there is a lack of intuition, due to the meaning behind the fact that a state or a path is accessible from the state where the commitment holds is not delineated. Particularly, an accessibility relation should specify the intuitive relation between a social commitment and accessible paths or states. Furthermore, an accessibility relation should clarify how in a computational model one can determine if a state or a path is accessible. The author then suggested different levels of verifiability in order to verify the correctness of the proposed semantics. Two verifiability levels are as follows: (1) 'every commitment to a putative fact can be verified or falsified by challenging that putative fact'; and (2) 'every commitment to a mental state can be similarly verified or falsified, but only through the more arduous route of eliciting the agent's beliefs and intentions'. As created and manipulated (or modified) commitments can be publicly recorded, the observer agent—as in Venkatraman and Singh (1999)—can be in turn exploited to test the compliance of other agents with a given protocol. However, when the semantics is given in terms of mental states, Wooldridge (2000) pointed out that it is very difficult to carry out the verification of this property as we do not understand how such states can be systematically attributed to agents' programs. Such a problem stems from the subjective semantics, which, in principle, refers to a mental component by arguing that interaction among agents should be sincere.

The first application of Singh's descriptive ontology of commitments (Singh, 1999) to design coordinated MAS was introduced by Xing and Singh (2001). They particularly proposed a set of commitment patterns inspired by design patterns to composedly model agent interactions. In this approach, each pattern captures an important scenario and can be specialized and applied to commitment actions. Different combinations of patterns can yield different kinds of agent interactions. In this approach, each pattern is

expressed as a CTL formula. The commitments are simply modeled as ADTs where the consequence is a predicate with a vector of domain arguments $\vec{v}$ to pass data values.

Example 17 *$C(Cus, Mer, NB, sendPayment(\overrightarrow{price, date}))$ means that Cus commits to Mer in the context of the NB protocol to send the payment, which is a combination of predicates about the price and date of the good item.*

Commitment actions are modeled as predicates (cf. Remark 3 to know why predicates are not suitable to represent commitments). The following formula expresses the relationship between the communication proposition *Inform* and action predicate *Create*:

$$\forall i, j, Pred, \vec{v} : A\Box\,[Inform(i, j, Pred(\vec{v})) \supset A\Diamond[Create\,(i, C\,(i, j, G, Pred(\vec{v})))]]$$

This formula means that along all paths in all states when agent i informs agent j about a predicate $Pred\,(\vec{v})$, then along all paths there is a possibility that i creates a commitment to bring about $Pred\,(\vec{v})$ toward j in the context G. The authors exploited a statechart to specify the behavior model of each agent. Indeed, the operational semantics provided by statecharts are exploited as a rigorous basis for coordinating the interactions of agents. To relate such operational semantics with temporal logic specifications, a CTL model is produced from agent's statechart. The soundness of this transformation is proved. The authors continued their approach in Xing and Singh (2003) by developing (1) an algorithm to transform the statechart of behavior model of agent into CTL model; (2) a library of commitment patterns; and (3) a library of behavior models of agents along with theorem proving which behavior models comply with which patterns. Notably, the agent communication semantics of this approach is formal, declarative, meaningful, and verifiable (using theorem proving). In theorem proving, the proof construction is in general difficult and requires a good deal of human ingenuity (Emerson, 1990). However, manual proof construction does not scale up well to large programs and fails 'to be of much help due to the inherent complexity of testing validity for even the simplest logics' (Clarke *et al.*, 1986). When commitments are considered as basic components in ACLs, no formal semantics is given because they are simply modeled as ADTs and treated as propositions in CTL model. Recall that a formal CTL-based framework is proposed to give only semantics for agent communication using commitments.

Mallya and Huhns (2003) and Mallya *et al.* (2004) developed an extension of CTL with (1) predicates to represent and reason upon commitments and related actions; and (2) two temporal quantifiers (existential and universal) to describe temporal deadlines in the form of time points and Allen's intervals. This extension provides a richer temporal aspect for the commitment consequences in order to capture real-life scenarios.

Example 18 *If the proposition p represents a price quote, then $[d_1, d_2]p$ means that p will be an offer for the period between d_1 and d_2. $C\,(Cus, Mer, [d_1 + d_2 + 24hours]p)$ states that Cus can commit to send the price p, which is only valid for an entire day, to Mer.*

To define the semantics of violation and fulfillment of the commitment c in question, the authors, respectively, introduced two predicates (*Breached* (c) and *Satisfied* (c)). For instance:

$$\begin{aligned} M, s \models Satisfied(c) \text{ iff } \Big(&\exists s_3 : s_3 \leq s \text{ and } M, s_3 \models Discharge(i, c) \text{ and},\\ &(\exists s_1 : s_1 < s_3 \text{ and } M, s_1 \models Create(i, c), \text{and}\\ &(\forall s_2 : s_1 \leq s_2 < s_3 \supset M, s_2 \models Active(c))\Big) \end{aligned}$$

The semantics of the *Satisfied* (c) predicate at s in a model M is defined in terms of whether or not the *Discharge* (i, c) of the commitment c that has been created in the past and still active holds. A commitment is active if it is not canceled, delegated, assigned, released, and discharged yet. The authors assumed that the discharge action 'brings about p, and conversely, if p occurs, the discharge action is assumed to have happened'. As a result, the discharge action's performance is in principle equivalent to the satisfaction of p. In fact, this assumption raises many issues. More specifically, Verdicchio and Colombetti (2004), in the same proceedings, criticized such an assumption by showing that the above semantics of the *Satisfied* (c) predicate appears 'bypassing the problem instead of solving it'. As the scope of the discharge action focuses on checking whether or not the truth condition of the commitment consequence holds, it in

turn suffers from the over-specification problem discussed in Singh (2008). To conclude, such a semantics straightforwardly supports formal, declarative, and meaningful criteria for agent communication but not the verifiable criterion. The authors also model commitments and associated actions as predicates, which in turn suffer from the problem discussed in Remark 3.

To enable a rich modeling of temporal deadlines for commitments, Torroni *et al.*, (2010) extended Mallya *et al.*'s (2004) proposal by using 'variables with domains' inside commitments. The authors showed that without such an extension, Mallya *et al.*'s representation of commitments does not cover some practical situations. For example, a commitment of *Cus* toward *Mer* to send the payment p is going to hold at a given moment in the interval beginning at t_1 and ending at t_2 would be represented in Mallya *et al.*'s model as: C (*Cus*, *Mer*, $[t_1, t_2]\,p$). This modeling enables reasoning about the temporal deadline without considering the p's meaning and specifying the time at which the commitment is satisfied. In Torroni *et al.*'s model, p is defined as a variable, which is bound to a domain interval: $[T]p$, $T \in [t_1, t_2]$. Thus, the commitment above can be written as:

$$C\,(Cus, Mer, [T]p),\ t_1 \leq T \leq t_2$$

When there exists a possible value of T in the range $[t_1, t_2]$, the commitment is satisfied and this value can be used for further inferences. This commitment is violated at time t (*viol* (C (*Cus*, *Mer*, $[T]\,p$, t))) 'due to the elapsing at time t of a time interval in which p was supposed to be verified'. The authors proposed a specification language called *commitment modeling language* (CML), which consists of a set of domain variables, constraints and rules. They used EC axioms not only for reasoning about the effects of commitment actions, but also for a static verification of properties and compliance checking, which tracks the evolution of commitment statuses at run time by making use of reactive event calculus (REC), which is implemented in SCIFF, an abductive proof procedure (Alberti *et al.*, 2008). In this approach, commitments are modeled as fluents and then there is no formal semantics for commitments (cf. Remark 1 for more information about the limitations of fluents). The proposed semantics for agent communication perspective is formal (where EC axioms are used to model commitment actions capturing the meaning of ACL messages), declarative, meaningful, and verifiable.

Gerard and Singh (2013) developed an analysis tool named 'Proton' to specify business protocols having social semantics and their refinement. In particular, Proton specifies a protocol declaratively in terms of (1) its agent roles; (2) guarded messages, which must be true before the messages can be sent by the roles; and (3) the meaning of each message as a set of actions applied to the roles' social states, holding atomic propositions that specify the states of commitments. Proton also describes the syntax for a refinement mapping. To verify refinement of protocols, they exploited model checking to compute whether a protocol refines correctly another one under a given mapping that contains the essential elements for refining any two protocols. Internally, their verification technique uses the Proton preprocessor that first reads the two protocols and mapping specifications and then generates both the ISPL model accepted by the MCMAS model checker (Lomuscio *et al.*, 2009) and CTL formulae expressing certain conditions. All generated CTL formulae must be satisfied in the generated model by MCMAS for the protocol refinement to hold. The proposed agent communication semantics is formal, declarative, meaningful, and verifiable. The formal semantics of commitments is entirely missing because the authors modeled commitments as objects with seven instances that are mapped into domain variables in the ISPL language. These variables cannot represent the real and concrete meaning of commitments. However, commitment actions are simply modeled as atomic propositions, not as effective actions. Technically, the operational semantics of discharge action—'which occurs implicitly when the consequent becomes true'—suffers from the over-specification problem.

Using the approach of Xing and Singh (2001, 2003), Telang and Singh (2009a, 2012) introduced an interesting business model that uses social commitments and agent-oriented concepts such as goals and tasks inspired by Tropos software engineering methodology (Bresciani *et al.*, 2004) to capture complex, long-lived business scenarios among business partners involved in service engagements (Telang & Singh, 2009a) or cross-organizational business processes that are the norms in today's MAS applications (Telang & Singh, 2012). As in Xing and Singh (2003), the authors developed a library of business patterns that in turn model recurring business scenarios wherein each pattern is delineated by a highly abstract-level based

upon the notion of commitments with some attributes: *name*, *intent*, *motivation*, *implementation*, and *consequence* inspired by classical object-oriented design patterns. In order to verify agent interactions, Telang and Singh proposed two different methods to implement this process. In the former one (Telang & Singh, 2009a), they introduced a reasoning algorithm, which in turn takes a business model populated by a set of business patterns and business interactions formalized using the UML sequence diagrams and returns a set of violated commitments. In the latter one (Telang & Singh, 2012), they exploited the NuSMV model checker (Cimatti *et al.*, 2002) to verify whether an operational model (a set of business interactions) correctly confirms the defined business model. Technically, a business model pattern is formalized as a set of CTL formulae. The following is an example of a business pattern and its CTL formalization.

Example 19 *When a commitment is inactive in the present state, then along all paths from the next state it could be inactive, active or detached. This would be expressed as a CTL formula as:* $A\Box$ *(Inactive* $\supset A \bigcirc$ *(Inactive* $\vee$ *Active* $\vee$ *Detached*)).

A commitment is simply modeled as an SMV module, which can be initiated as a domain variable in the main module. To evaluate this semantics for agent communication against our criteria, it is formal (where a CTL-based framework is introduced to express business model patterns), declarative, meaningful, and verifiable. However, there is no formal semantics for commitments as they are modeled simply as SMV modules. In addition, formal semantics for commitment actions themselves are not considered. With respect to the flexibility of classical commitment machines (Yolum & Singh, 2002a), adopting the UML sequence diagram to model the behavior of interacting agents forces the temporal ordering of action executions, which in principle looses the flexibility supported by the adoption of commitments as shown in Baldoni *et al.* (2010, 2011).

5.2 Bentahar and colleagues

El-Menshawy *et al.* (2011b) presented a framework that comprises of three parts aiming at excluding spurious aspects that plague some of the above proposals (e.g. lacking intuition and computation, modeling commitments as fluents, propositions and predicates, and reducing commitments into domain variables). In the first part, they introduced a new temporal logic, named CTLC, an extension of CTL with modality for commitments. They defined a social accessibility relation R_{sc} to interpret the semantics of the commitment modality. This is the first approach that associates the formalism of interpreted systems introduced in Fagin *et al.* (1995) to model MASs with the Kripke model in order to compute the accessibility relation and define an intuitive semantics of social commitments, which is missing in existing proposals of agent communication models. In the formalism of interpreted systems, each agent $i \in \mathcal{A}$ is characterized by a set of local states L_i, a set of local actions, a local protocol, and a local evaluation function. The set of all global states S is a subset of the Cartesian product of all local states of n agents at a given time: $S \subseteq L_1 \times L_2 \times \ldots \times L_n$. The standard CTL model $M = (S, R_t, V, I)$ is extended to $M' = (S, R_t, R_{sc}, V, I)$, where $R_{sc} : S \times \mathcal{A} \times \mathcal{A} \rightarrow 2^S$ is the social accessibility relation for commitments. It is defined as follows:

$$s' \in R_{sc}(s, i, j) \textit{ iff } \exists \overline{s} \in S : l_i(s) = l_i(\overline{s}) \textit{ and } l_j(\overline{s}) = l_j(s')$$

where $l_i(s)$ denotes the local state of agent i in the global state s. Intuitively, s' is accessible from s, that is $s' \in R_{sc}(s, i, j)$ iff there is an intermediate state $\overline{s}$, so that there is no difference for the debtor i between being in s and $\overline{s}$; but for the creditor j there is no difference between being in the intermediate state $\overline{s}$ and accessible state s'. In the second part, the semantics of commitments is formally defined as follows:

$$M', s \models C(i, j, \varphi) \textit{ iff } \forall s' \in S, \textit{ if } s' \in R_{sc}(s, i, j), \textit{ then } s' \models \varphi$$

which means that the consequence φ is true in every accessible state from the current state computed using $R_{sc}(s, i, j)$. In the last part, the authors show how the problem of model checking CTLC can be formally reduced, using a transformation function $\mathscr{F}$, into the problems of model checking CTLK (an extension of CTL with knowledge modality (Penczek & Lomuscio, 2003)) and ARCTL (an extension of CTL with action formulae (Pecheur & Raimondi, 2007)) in order to be able to, respectively, use the MCMAS and

extended NuSMV (Lomuscio *et al.*, 2007). The more *interesting* point in this approach is that the commitment modality is reduced into the knowledge modality, not into domain variables, as follows:

$$\mathscr{F}\left(C\left(i, j, \varphi\right)\right) = \widehat{K}_i \mathscr{F}(\varphi) \wedge \mathrm{E} \bigcirc \widehat{K}_j \mathscr{F}(\varphi)$$

where $\widehat{K}_i\varphi$ read as 'possible knowledge' of agent i (Fagin *et al.*, 1995), is the dual of agent knowledge $K_i\varphi$ (i.e. $\widehat{K}_i\varphi \triangleq \neg K_i \neg \varphi$) and $E \bigcirc \widehat{K}_j$ is read as there is a path such that in the next state agent j possibly knows φ. The authors also implemented the reduction technique on top of the MCMAS and extended NuSMV to automatically verify a business protocol against some desirable properties, such as *reachability, liveness,* and *safety*.

Example 20 *Reachability property: Given a particular state, is there a valid computation sequence to reach that state from an initial state. The following formula is used to check that the Payment state is reachable from the Goods state in NB:* $E(\neg Goods\ U\ (Goods \wedge C(Cus, Mer, sendPayment)))$.

It is clear that our seven criteria for agent communication semantics using commitments are successfully met. However, the authors do not consider the semantics of commitment actions and their verification.

To address the identified limitations, El-Menshawy *et al.* (2011c) extended CTLC introduced in El-Menshawy *et al.* (2011b) with two modalities to represent and reason about fulfillment (*Fu*) and violation (*Vi*) of commitments. Given that, the authors proceed to develop a new symbolic algorithm to perform the model checking of the proposed logic and fully implemented such an algorithm on top of the MCMAS model checker developed for MASs (Lomuscio *et al.*, 2009). An example of fulfillment is as follows.

Example 21 $E\Diamond\ Fu(C(Cus, Mer, sendPayment))$*, meaning that there is a path in its future Cus fulfills its commitment by sending the promised payment to Mer.*

Bentahar *et al.* (2012) and El-Menshawy *et al.* (2013b) redefined both the social accessibility relation introduced in El-Menshawy *et al.* (2011c) to account for the intuition that social commitments are conveyed through communication between agents and the semantics of commitments and their fulfillments. They also removed the violation modality presented in El-Menshawy *et al.* (2011c); instead the violations of commitments are expressed as properties to simplify the proposed semantics.

Example 22 *The violation property is valid when there is a computation so that in its future a commitment to deliver the requested goods is established but from the moment where the commitment is active there is a possible computation where globally the fulfillment never happens:* $E\Diamond(C(Mer, Cus, deliverGoods) \wedge E\Box(\neg Fu(C(Mer, Cus, deliverGoods))))$.

This refinement is extremely important as far as time complexity and space complexity are concerned. The refined logic is called CTLC$^+$. As in El-Menshawy *et al.* (2011c), Bentahar *et al.* (2012) adopted the direct verification technique that develops symbolic algorithms needed for the new modalities. The verification technique introduced by El-Menshawy *et al.* (2013b), on the other hand, is based on formally reducing the problem of model checking CTLC$^+$ into the problem of model checking ARCTL and the problem of model checking GCTL* (Bhat *et al.*, 2001) (a generalized version of CTL* with action formulae) so that the extended NuSMV symbolic model checker and the CWB-NC automata-based model checker[7] as a benchmark are usable. The authors tested the satisfiability of a set of desirable properties, such as *safety* and *reachability*.

Example 23 *The safety property entails 'something bad never happens'. It can be generally expressed by* $A\Box\neg p$*, where p characterizes a 'bad' situation, which should be avoided. The bad situation happens when Mer fulfills its commitment by delivering the requested goods, but Cus never commits to send the payment:* $A\,\Box\neg(Fu(C(Mer, Cus, deliverGoods) \wedge A\Box\neg\ C(Cus, Mer, sendPayment)))$.

El-Menshawy *et al.* (2013b) also showed that their formal reduction technique is easy to implement and allows one to compare different verification methods and techniques with respect to the same logic. To come into the evaluation of Bentahar *et al.* (2012) and El-Menshawy *et al.* (2013b), the semantics

[7] `http://www.cs.sunysb.edu/cwb/`

introduced in these proposals evidently meets our seven criteria. However, there is no semantics for other commitment actions and conditional commitments. Another issue is that the semantics of fulfillment implies that the commitment is active at the moment of its fulfillment, that is $Fu(C(i, j, \varphi)) \supset C(i, j, \varphi)$, which is not commonly accepted in the literature (cf. e.g. Yolum & Singh, 2004; Winikoff *et al.*, 2005; Singh, 2008).

5.3 Yolum and colleagues

Kafali *et al*. (2014) developed a tool, called $\mathcal{PROTOSS}$, to detect and predict possible privacy violations within online social networks (OSNs). The privacy agreements that exist among each user and the OSN operator and the relations among the users constitute the formal model. Such privacy agreements are represented as a set of commitments. The tool has a semantic reasoning component that makes use of an ontology, which is used to refine commitments. For instance, if the OSN operator commits to a user not to share her information, then via reasoning on the ontology, the system can infer that neither a person's location nor her pictures can be shared. Moreover, undefined privacy concerns can be discovered through the ontology. The following example shows a commitment representing a privacy agreement.

Example 24 *CC* (*operator*, *charlie*, *colleague* (*charlie*, *X*), ¬*shareLocation* (*charlie, X*)), *where the social network operator (operator) is the debtor, Charlie is the creditor, the antecedent colleague(charlie, X) represents charlie and some another user X are colleagues and the consequent* ¬*shareLocation* (*charlie*, *X*) *represents that the location information of charlie is shared with X.*

The above example shows that the operator will be committed not to share Charlie's location information with his colleagues if Charlie declares an individual as his colleague. After these agreements are specified, then the OSN operator is asked to check if there are any privacy violations in the system model. Following the above example, if a colleague ends up seeing Charlie's location, this would yield a privacy violation.

Kafali *et al.* represent privacy violations as commitment violations in the system model and employ model checking to detect such violations. In general, the model checking technique is used to check that the private behavior rules—which define the operational behavior of the OSN operator—comply with privacy agreement shared between a user and the OSN operator, such that the operator would act in a way that honors its agreement with the user. Specifically, the authors use NuSMV as the underlying model checker and model a commitment as an SVM module in order to enable NuSMV to deal with commitments, in the same spirit of Telang and Singh (2012). The commitment module defines the statuses of a commitment, which correspond to the commitment states. As the focus is on the privacy violations, only four commitment states are implemented: conditional, active, fulfilled, and violated. The life cycle implemented there allows a commitment to be fulfilled only if both the antecedent and consequent hold, as if the consequent holds without the antecedent, there is still a privacy violation. Using the commitment module, domain variables of commitment types can be specified.

With the help of $\mathcal{PROTOSS}$, which is implemented as the privacy checker for the OSN operator, privacy properties are expressed in CTL and then automatically checked. For example, the following property describes if Charlie and Linus are colleagues, the commitment introduced in Example 24 (say $c1$) is violated. The model checking then decides if this is true, thereby leading a privacy violation.

$$A\Box(colleague\_charlie\_linus \supset A\Diamond c1.status = VIOLATED)$$

The workings of the approach have been tested over privacy scenarios and their performance have been evaluated on a Facebook data set. Their results show that when the social network is small, the commitments can be verified in a reasonable time, but when the network grows, the model needs to be pruned to be verified efficiently. As Kafali *et al.*'s proposal follows the same methodology as Telang and Singh (2012) when it comes to the representation of commitments as SMV modules, the two proposals share the same evaluation with regard to our criteria.

Table 2 summarizes our results of evaluating current proposals that use CTL (or an extension of CTL) to define the semantics for ACL messages using commitments and related concepts.

We conclude this section with Table 3, which in principle presents the empirical results reported in the proposals reviewed in Table 2 in order to be able to evaluate the performance of their employed verification methods in terms of number of agents, number of reachable states, and execution time.

Table 2 Summary of computation tree logic and commitment-based agent communication

	Agent communication semantics				Commitment		
	For.	Dec.	Mea.	Ver.	Mod.	Sem.	Verification method
Venkatraman and Singh (1999)	√	√	√	√	Abstract data type		Model checking-like
Singh (2000)	√	√	√	√	Temporal modality	√	Model checking-like
Xing and Singh (2001, 2003)	√	√	√	√	Abstract data type		Theorem proving
Mallya and Huhns (2003, 2004)	√	√	√		Predicate		
Gerard and Singh (2013)	√	√	√	√	Object		Model checking
Telang and Singh (2012)	√	√	√	√	Module		Model checking
El-Menshawy *et al.* (2011a)	√	√	√	√	Temporal modality	√	Model checking (formal translation to MCMAS and NuSMV)
El-Menshawy *et al.* (2011b)	√	√	√	√	Temporal modality	√	Dedicated a new model checking algorithm
Bentahar *et al.* (2012)	√	√	√	√	Temporal modality	√	Dedicated a new model checking algorithm
El-Menshawy *et al.* (2013b)	√	√	√	√	Temporal modality	√	Model checking (formal translation to extended NuSMV and CWB-NC)
Kafali *et al.* (2014)	√	√	√	√	Module		Model checking

Table 3 Empirical results reported in the proposals reviewed in Table 2

		Number of agents	Number of states	Time (seconds)
Gerard and Singh (2013)	NetBill2 protocol	3		1.2
Telang and Singh (2012)	Quote to cash process	6	12 600	0.1
El-Menshawy *et al.* (2011b)	NetBill protocol	6	239	<0.01
		20	4.59 517e + 08	1128
Bentahar *et al.* (2012)	Insurance claim processing	48	6.85 373e + 12	14 477
El-Menshawy *et al.* (2013b)	NetBill protocol	6	238 787	630.914

By comparing the direct verification technique in El-Menshawy *et al.* (2011c) and the reduction verification technique in El-Menshawy *et al.* (2013b) when the number of agents is six agents, we found that

1. The direct verification technique needs < 0.01 second to execute the verification process, whereas the reduction verification technique needs 630.914 seconds (cf. Table 3). The reason why the reduction technique is slower because its execution time is the summation of the time needed to carry out the actual verification process and the time needed to carry out the reduction preprocessing step.
2. The reduction verification technique is hard to scale up in order to include more agents (six agents in El-Menshawy *et al.* (2013b) and 20 agents in El-Menshawy *et al.* (2011c)), because this technique is based on transformation rules that require extra states (Bentahar *et al.*, 2010) and transitions (El-Menshawy *et al.*, 2011a, 2013a) to capture the semantics of certain actions as well as additional temporal operators, which are added to find the equivalent formulae to the transformed ones.
3. The number of reachable states in the reduction verification technique (238 787 states, see Table 3) is more than the corresponding one in the direct verification technique (239 states in El-Menshawy *et al.* (2011c)), because some states are added in the transformation process.

6 Full computation tree logic and commitment-based agent communication

CTL* extends CTL by allowing basic temporal operators in which the path quantifier (E or A) can prefix an assertion composed of unrestricted combinations (i.e. involving arbitrary nestings and Boolean

connectives) of the linear time operators $\Diamond$, $\Box$, $\bigcirc$, and U. It was proposed as a unifying framework in Emerson and Halpern (1986), subsuming CTL and LTL. Recall that the underlying nature of time in CTL* is branching: each moment in time might split into alternative courses representing different possible futures, which is suitable for reasoning about non-deterministic and concurrent programs. Technically, in CTL*, one can distinguish between two types of formulae: state formulae and path formulae. The state formulae—true or false of states—are assertions about the atomic propositions in the states and their branching structure, whereas path formulae—true or false of paths—express temporal properties of paths.

DEFINITION 5 *Given the set $\mathcal{PV}$ of atomic propositions, the syntax of CTL* formulae is given by a BNF grammar as follows (Clarke et al., 1999):*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid E\psi$$
$$\psi ::= \varphi \mid \neg\psi \mid \psi \vee \psi \mid \bigcirc\psi \mid \psi\, U\, \psi$$

where φ is a state formula and ψ is a path formula. Other Boolean connectives and temporal operators can be introduced as abbreviations in the usual way.

A formula of CTL* is interpreted with respect to a transition system $M = (S, R_t, V, I)$ where S, R_t, V, and I have the same meaning as in LTL.

DEFINITION 6 *We write $M, s \models \varphi$ (respectively, $M, \pi \models \varphi$) to mean that a state formula φ (respectively, a path formula φ) is true in a transition system M at state s (respectively, along a path π). We define the relation $\models$ inductively as:*

- $M, s \models p$ *iff* $p \in V(s)$
- $M, s \models \neg\varphi$ *iff* $M, s \not\models \varphi$
- $M, s \models \varphi \vee \psi$ *iff* $M, s \models \varphi$ *or* $M, s \models \psi$
- $M, s \models E\varphi$ *iff* *there exists a path π such that $\pi(0) = s$ and* $M, \pi \models \varphi$
- $M, \pi \models p$ *iff* $M, \pi(0) \models p$
- $M, \pi \models \neg\psi$ *iff* $M, \pi \not\models \psi$
- $M, \pi \models \varphi \vee \psi$ *iff* $M, \pi \models \varphi$ *or* $M, \pi \models \psi$
- $M, \pi \models \bigcirc\psi$ *iff* $M, \pi_1 \models \psi$
- $M, \pi \models \varphi\, U\, \psi$ *iff* *for some $k \geq 0$ we have* $M, \pi_k \models \psi$ *and* $M, \pi_i \models \varphi\, \forall\, 0 \leq i < k$

The state formula $E\varphi$ is satisfied in M at s iff there exists a path π starting at s that satisfies φ. In addition, the state formulae can be interpreted on paths such that M along a path π satisfies a state formula φ iff M satisfies φ at the initial state of the path: $M, \pi \models \varphi$ *iff* $M, \pi(0) \models \varphi$. $\bigcirc\psi$ holds on a path π when ψ is satisfied on the path starting from the next state of the path π, and $\varphi\, U\, \psi$ holds on a path if φ holds on the path until ψ becomes true.

In the rest of this section, CTL* is extended with modalities or predicates to mainly define semantics of speech-act-based ACL with respect to social notions. The resulting logical model is exploited to derive a new specification language of commitment-based protocols.

6.1 Colombetti and colleagues

Verdicchio and Colombetti (2003, 2004) introduced a logical framework for the definition of ACL semantics based on the concept of social commitments resulting from the performance of speech acts by presenting a branching-time temporal logic, named CTL$^{\pm}$, which in turn enriches CTL* with (1) 'past-directed temporal operators'; and (2) meta-language predicates (or *many-sorted first-order language*) containing terms, which in turn denote events, actions, commitments, and commitment actions. They also introduced a number of appropriate axioms to describe the constitutive effects of the action types for commitment manipulations. In other terms, each axiom describes the state of affairs that necessarily holds once a token of a given action type is successfully carried out. There are basically two possibilities to represent the consequence of commitment: (1) it can be represented as a CTL$^{\pm}$ formula, in which case, a commitment can be modeled as a modal operator; and (2) it can be represented as a first-order term, in which case, a commitment can be modeled as a first-order formula. The authors forcefully argued that modeling a commitment as a first-order formula rather than a modal operator has the advantage that the

'technicalities required by a predicative representation are simpler than the ones required by a modal representation'. Such a first-order term can be viewed as the representation of a concrete *Content Language* (CL) statement. To achieve this goal, the syntax of CL statement can be represented as a first-order term in $CTL^{\pm}$. Technically, such a first-order term, say φ, is represented as a $CTL^{\pm}$ formula using a 'truth-persevering translation' $\lfloor\varphi\rfloor$. However, modeling commitments using many sorted first-order terms waives the formal semantics and the sense of formulae, which in turn makes using model checking-based verification inappropriate. Inspired by Reichenbach's (1947) terminology, the semantics of fulfilling commitments is defined as follows:

$$Fulf(e, i, j, \varphi) \triangleq C(e, i, j, \varphi) \wedge A\Diamond^{-}(Happ(e) \wedge \lfloor\varphi\rfloor)$$

where $\Diamond^{-}$ is read as 'sometimes in the past'. The predicate $Happ(e)$ states that an event e has happened and $\lfloor\varphi\rfloor$ is a truth-preserving translation of the commitment consequence φ into a $CTL^{\pm}$ formula. The semantics of fulfillment means that the commitment is fulfilled at s' (Reichenbach's point of event), when the *commitment-inducing event e* has been made at s (Reichenbach's point of speech) where the truth value of $Happ(e)$ and $\lfloor\varphi\rfloor$ are true 'at some state in the past of s' on every path starting from s and going throughout s'''. Notably, this agent communication semantics meets the following criteria: formal, declarative, and meaningful. However, the semantics of fulfillment action in principle suffers from the over-specification problem as it considers the truth value of the commitment consequence, which is the main component of the semantics of commitment itself. The authors also do not consider the verification issue. From our point of view, the verification of this logic by means of model checking is very hard to perform because: (1) the proposed logical model uses superfluous translation of the commitment consequence and past operator that points to the state at which the commitment is made; and (2) the model checking of predicate logic is undecidable as we cannot write a procedure that works for all φ (the proof of this claim is introduced in Huth & Ryan (2004)). Such a temporal language has been extended with two interval operators, dates, and times to express a 'rich assortment of temporal conditions', such as deadlines in a natural way, which is missing in the CL expressions of FIPA-ACL (Verdicchio & Colombetti, 2005). Here are some examples of temporal deadlines:

- The agreed payment is due before the end of November 2014:
  $\langle cd(), 2014Y11M\rangle \bullet Done(Payment)$, where $cd()$ is the constant of the current time.
- The agreed payment is due within 45 days from the time of delivery:
  $\langle Done(delivery), cd() + 45D\rangle Done(payment)$.
- The auction will be open for 40 hours:
  $[Done(openAuction), cd() + 40H]Open(auction)$.

This approach could be considered as a generalization of the work introduced in Mallya *et al.* (2004). Colombetti *et al.* (2004) and Verdicchio and Colombetti (2006) showed that FIPA's communicative act library can be effectively redefined using a commitment-based semantics instead of FIPA's mental semantics with respect to their logical model introduced first in Verdicchio and Colombetti (2003) and extended in Verdicchio and Colombetti (2004). For example see the following.

Example 25 *Basic communicative acts (e.g. inform, request, agree, propose, accept-proposal, refuse, reject-proposal, and cancel) are mapped onto message types and commitment manipulation act types; for instance, the inform act is defined as creating a commitment by performing a make-commitment act.*

6.2 Bentahar and colleagues

The first work on combing social commitments as deontic notions and arguments into the paradigm of ACL was done by Bentahar *et al.* (2003). In this hybrid approach, the social and public aspects of conversational agents are captured by commitments and the reasoning aspects are delineated by means of arguments. The authors claimed that existing approaches introduced to define ACL semantics are 'not exclusive but rather complementary'. This approach, however, waives defining formal semantics of commitments, arguments, and semantic link between them. In continuation of this work, Bentahar *et al.* (2004a, 2004b, 2007) adopted a temporal logic to address such limitations. In particular, they extended

CTL* with (1) modalities for commitments and two-party actions; (2) argument modality; and (3) a DL, which in turn captures the actions that agents are committed to achieve. They then introduced two accessibility relations to define the semantics of commitment and argument modalities. For instance, the accessibility relation dedicated to commitments (Bentahar *et al.*, 2007) is defined as: $R_{sc} : \mathcal{A} \times \mathcal{A} \times S \rightarrow 2^{\Pi}$, which in turn associates with a state s a set of accessible paths along which an agent commits toward another agent. Thus, the semantics of the commitment modality is given as:

$$M, s \models C(i, j, \varphi) \textit{ iff } \forall \pi : \pi \in R_{sc}(s, i, j), M, \pi \models \varphi$$

The commitment formula is satisfied in a model M at s iff the consequence φ is true along every accessible path started at s and computed by R_{sc}. The semantics of the *Satisfy* (discharge) action is defined in terms of whether the commitment has been created in the past and still active and its consequence holds or not. This would be expressed as:

$$\begin{aligned} M, \pi(s) \models \textit{Satisfy}(i, C(i, j, p)) \textit{ iff } \quad & M, \pi(s) \models \textit{Active}(C(i,j,p)) \textit{ and } \exists s' : T(s') \leq T(s) \\ & \textit{and } M, s' \models \textit{Create}(i, C(i,j,p)) \textit{ and } M, \pi(s) \models p \end{aligned}$$

where $T(s)$ gives the time point associated to the state s. A commitment is active iff this commitment was already created, and till the present moment, the commitment in question was not withdrawn:

$$M, \pi(s) \models \textit{Active}(i, C(i,j,p)) \textit{ iff } M, \pi(s) \models \neg \textit{Withdraw}(i, C(i,j,p)) \, U^{-} \textit{Create}(i, C(i,j,p))$$

where U^{-} is interpreted as 'since in the past'. To define flexible conversations, the authors interpreted a speech act not as an action performed on a commitment as usual, but also on a commitment consequence to enable agents to defend and justify their commitment consequence and to attack and challenge the commitment consequence of other agents. This semantics for agent communication meets formal, declarative, and meaningful criteria. As we discussed in Colombetti (2000) and Singh (2008), the intuition of the accessibility relation and how accessible paths are practicality computed is not clear. Furthermore, this semantics is delineated by a recursive manner (i.e. the semantics of one action depends recursively on the semantics of one or more other actions), which in turn induces its model checking procedure quite hard. In addition, the semantics of satisfy action suffers from the over-specification problem as it focuses upon checking the truth condition of the commitment consequence.

El-Menshawy *et al.* (2009b, 2010b) addressed the limitations of Bentahar *et al.* (2004a, 2004b, 2007) by developing a logical model based on a new temporal logic, named CTL^{*sc}, which in principle extends CTL* with (1) past-directed temporal modalities; and (2) modalities for commitments and all pertained actions. The proposed logic is exploited to drive a new specification language of commitment-based protocols to particularly define the meaning of protocol messages in terms of commitments. In particular, the semantics of each action does not depend upon other actions using the notion of accessible and non-accessible paths. For instance, the semantics of fulfillment (discharge) action is satisfied in a model M along a path π_i starting at s_i iff (1) the commitment was established in the past at s_j (after performing the creation action) through the prefix of π starting from s_j denoted by $\pi_i \downarrow s_j$; (2) all paths starting at state s_{i+1} (the state resulting from the fulfillment action) using R_{sc} at state s_j (where the commitment has been established) are accessible paths; and (3) at the present state s_i, there exists a possible choice of not fulfilling the commitment as the prefix $\pi_i'' \downarrow s_j$ of a non-accessible path π_i'' emanating from s_i is present. This would be expressed as:

$$\begin{aligned} (M, s_i, \pi_i) \models \textit{Fulfill}(i, C(i,j,\varphi)) \textit{ iff } \; & (1) \exists j \leq i : (M, s_j, \pi_i \downarrow s_j) \models C(i,j,\varphi) \textit{ and} \\ & (2) (s_i, \textit{Fulfill}, s_{i+1}) \in R_t \textit{ and} \\ & \quad \forall \pi'_{i+1} \in \Pi^{s_{i+1}} : \pi'_{i+1} \downarrow s_j \in R_{sc}(s_j, i, j) \textit{ and} \\ & (3) \exists \pi_i'' \in \Pi^{s_i} : \pi_i'' \downarrow s_j \notin R_{sc}(s_j, i, j) \end{aligned}$$

where Π^{si} is the set of paths starting at s_i. Thus, the semantics of fulfillment action is cured from the over-specification problem raised in Verdicchio and Colombetti (2003), Bentahar *et al.* (2004a, 2004b), Mallya *et al.* (2004), Verdicchio and Colombetti (2004), Bentahar *et al.* (2007). The idea is that 'being accessible means that the consequence φ is true along all the paths $\pi'_{i+1} \downarrow s_j$ and fulfillment occurs automatically when the commitment consequence holds'. They also introduced a new definition of assignment and delegation actions by considering the relationship between the original and new

commitment consequences, which is missing in the current approaches. The authors expressed in CTL*^{sc} a set of desirable properties named 'functional correctness', such as *liveness*, *reachability*, *safety*, and *fairness constraint*. An example of the reachability property is given as follows.

Example 26 *Along all paths where Cus is always in a request state, a certain state of delivering goods by Mer is eventually reachable:* $A\Box^{+}$*(Request* $\supset E\Diamond^{+}$ *Goods). The negation of the reachability property could be used to show the absence of deadlock in the protocol.*

The authors also proposed a symbolic verification technique based on informally reducing the problem of model checking CTL*^{sc} into the problems of model checking LTL^{sc} and CTL^{sc} inspired by the notion introduced in Clarke *et al.* (1999) regarding the reduction of model checking CTL* into the problems of model checking LTL and CTL. Notice that LTL^{sc} and LTL^{sc} are the standard LTL and CTL extended with commitments and their actions. In this technique, the participating agents in the protocol are defined either as modules using SMV (the input language of NuSMV) or as sets of local states and actions along with local protocol, and local evaluation function using the ISPL language. However, modeling commitments and their actions as domain variables in the SMV and ISPL languages waives real and concrete meanings of commitments and their actions. El-Menshawy *et al.* (2013a) recently addressed this limitation by developing a reduction technique that formally transforms the problem of model checking CTL*^{sc} into the problem of model checking GCTL*, so as to use the CWB-NC model checker. An example of the liveness property checked in this proposal is given as follows: the liveness property means that 'something good will eventually happen'. For example:

Example 27 *Along all paths it is globally the case that if Mer fulfills its commitment by delivering the promised goods, then in all computations in the future, Cus will either (1) fulfill the commitment by sending the payment; (2) withdraw this commitment; or (3) violate it:*

$$A\Box(EFu(Mer, C(Mer, Cus, deliverGoods)) \supset A\Diamond[EFu(Cus, C(Cus, Mer, sendPayment)) \vee EWi(Cus, C(Cus, Mer, sendPayment)) \vee EVi(Cus, C(Cus, Mer, sendPayment))])$$

However, as we discussed in El-Menshawy *et al.* (2013b) and proved in Bentahar *et al.* (2012), formal reduction techniques show very limited scalability and high memory usage. The conditional commitment semantics introduced in El-Menshawy *et al.* (2013a) has a limitation resulting from using the material implication operator to define the causal relationship between conditional and unconditional commitments: $CC(Mer, Cus, sendPyament, sendReceipt) \triangleq sendPayment \supset C(Mer, Cus, sendReceipt)$. This limitation means that the conditional commitment *CC* (*Mer*, *Cus*, *sendPyament*, *sendReceipt*) could become active without satisfying its antecedent *sendPayment*. This limitation also appeared in Pham and Harland (2007). To conclude these approaches, their semantics are formal, declarative, meaningful, and verifiable and at the same time there is a formal semantics for commitments and all associated actions (El-Menshawy *et al.*, 2013a).

Bentahar *et al.* (2009, 2010) presented a verification technique based on informally reducing ACTL* (an extension of CTL* with modalities for commitments, arguments, and commitment actions) and protocols into Alternating Büchi Tableau Automata so that they made use of the CWB-NC model checker in order to address the verifiability criterion, which is the main limitation of their approaches (Bentahar *et al.*, 2004a, 2004b, 2007). However, this approach is still suffering from lacking formal semantics for commitments and their actions as commitments are defined as simple variables and agent actions as atomic propositions using CCS (the input language of the CWB-NC). In addition, they only considered two-party actions. Table 4 summarizes our results of investigating the proposals that adopt CTL* (or an extension of CTL*) to (1) represent and reason about commitments; (2) define the semantics for ACL messages using commitments; and (3) specify commitment-based protocols and express their properties.

We conclude this section with Table 5, which introduces the empirical results reported in some of the reviewed proposals in Table 4, so as to evaluate the performance of their employed verification methods in terms of number of agents, number of reachable states, and execution time. From Table 5, the

Table 4 Summary of full computation tree logic and commitment-based agent communication

	Agent communication semantics				Commitment		
	For.	Dec.	Mea.	Ver.	Mod.	Sem.	Verification method
Verdicchio and Colombetti (2003, 2004)	√	√	√		Predicate		
Bentahar *et al.* (2003, 2004a, 2007)	√	√	√		Temporal modality	√	
Bentahar *et al.* (2009, 2010)	√	√	√	√	Temporal modality	√	Model checking (informal translation to MCMAS and CWB-NC)
El-Menshawy *et al.* (2009b, 2010b)	√	√	√	√	Temporal modality	√	Model checking (informal translation to MCMAS and NuSMV)
El-Menshawy *et al.* (2013a)	√	√	√	√	Temporal modality	√	model checking (formal translation to CWB-NC)

Table 5 Empirical results reported in the proposals reviewed in Table 4

		Number of agents	Number of states	Time (seconds)	Tool
Bentahar *et al.* (2010)	NetBill protocol	2	2851	0.5	MCMAS
	NetBill protocol	2	2593	0.484	CWB-NC
El-Menshawy *et al.* (2013a)	NetBill protocol	8	428 545	5123.356	CWB-NC
	Contract Net protocol	9	4537	4.103	CWB-NC

authors El-Menshawy *et al.* (2013) considered more interacting agents (nine agents). In fact, we could not compare between these proposals (Bentahar *et al.*, 2010; El-Menshawy *et al.*, 2010b, 2013a) because they employ different (1) logical models; (2) semantics for commitments and their fulfillments; (3) temporal modalities that were used to extend CTL*; and (4) temporal properties that are being model checked. In the table, it is clear that the results of symbolic techniques (MCMAS and NuSMV) in terms of execution time are better than the corresponding ones of the automaton-based technique (CWB-NC).

7 Other logical languages

As aforementioned in the introduction, commitment-based protocols are flexibly specified outside the agents and independently of their architecture with regard to creation, manipulation, and satisfaction of commitments among such interacting agents. Yolum and Singh (2002a) developed a formalism named 'commitment machines' to represent, model, reason upon, and execute commitment-based protocols using a CTL-like semantics introduced by Singh (2000). The main idea of the commitment machine is to label states with commitments as well as some literals (if any) holding in those states, whereas transitions among states are labeled with actions applied to these commitments. Such a formalism is enabling agents to logically reason about their actions allowable in the protocol to compute their 'legal computations'. Fornara and Colombetti (2002) demonstrated that commitments lend themselves to operationalization in a more traditional way. As we understood, this is the idea of compiling a commitment machine into a traditional representation such as a FSM over finite computations, so that 'the desired effect can be obtained without representing and reasoning about declarative meanings at run time' (Yolum & Singh, 2002a). In other terms, such compilation deletes the opportunities for flexibility, which in principle characterizes a commitment representation. Furthermore, Yolum and Singh (2002a) pointed out that FSMs resulting from compiling commitment machines are still beneficiary for interacting agents who waive the ability to reason logically.

Winikoff *et al.* (2005) exploited the EC to revise some of Yolum and Singh's proposals, but without considering the compilation of commitment machines into FSMs. In particular, they improved the methodology by which (1) commitments are discharged in certain situations, such as nested conditional commitments and symmetrical; and (2) pre-conditions are specified. Furthermore, Winikoff (2007) extended the framework of commitment machines to flexibly model interactions among autonomous agents in distributed systems. The new version is named 'distributed commitment machines'. The authors then went forward to explore the properties of distributed and centralized commitment machines and to show how the properties of distributed commitment machines can be exploited to provide 'a simple implementation' of commitment machine-based interactions in distributed settings. Recently, Singh (2007) generalized the formalism of commitment machines introduced in Yolum and Singh (2002a) and Winikoff *et al.* (2005) to include 'non-terminal protocols' (or those that have cycles) analogous to those in real-life business applications. Singh's commitment machine is compiled into Büchi automaton over infinite computations (i.e. its acceptance condition is infinite). Hereafter, we briefly present two approaches that use an action logical language to specify commitment machines.

7.1 Action logical language: event calculus

Yolum and Singh (2002b, 2004) specified commitment-based protocols using commitment machines with the use of a subset of Shanahan's EC (Shanahan, 1997). Roughly speaking, EC is a logical language for representing and reasoning upon actions and their effects. The basic components of EC are *fluents* (properties or atomic propositions holding within time intervals) and *events* (actions happening at certain time points) (Shanahan, 1997). Fluents are initiated and terminated by occurring events (i.e. events manipulate and modify fluents). In this approach, EC is akin to 'many-sorted first-order predicate calculus' with eight predicates and a set of axioms to represent and in turn reason upon actions wherein commitments are modeled as predicates (cf. Remark 3 about the problem resulting from using predicates). For example, the discharge action, whose purpose is to successfully fulfill a commitment, is axiomatically defined as:

$$terminates(E(I), C(I, J, P), T) \leftarrow happens(E(I), T) \wedge discharge\,(E(I), C(I, J, P))$$

This axiom means that when an event $E(I)$ has been carried out by I at time T and the commitment is successfully discharged, the active commitment is stopped (or terminated). Such a discharge axiom can be delineated by another way (cf. Winikoff *et al.* (2005) framework, fig. 7) as:

$$terminates(E(I), C(I, J, P), T) \leftarrow$$
$$holds\_at(C(I, J, P),\ T) \wedge\ happens(E(I), T) \wedge initiates(E, P', T) \wedge subsumes(P', P)$$

The new axiom focuses on checking the status of commitment in question at time T and the existence of a new fluent P' initiated by occurring of an event $E(I)$ at the same time and in turn subsumed the commitment consequence P, and if so, the commitment is terminated. Notice that the predicate *subsumes* checks implicitly whether or not P holds (for other axioms that define other commitment actions, see Winikoff *et al.* (2005)). Yolum and Singh's approach (Yolum & Singh, 2002b, 2004) for defining the semantics of agent communication meets formal, declarative, and meaningful criteria. Moreover, the authors exploited an abductive EC planner (Shanahan, 2000) to logically compute reasoning queries with respect to the EC axioms (in the form of reasoning rules) that specify protocols: Given the initial protocol state, final protocol state, and protocol specification, the reasoner (planner) in turn computes 'all possible protocol runs' (where a run is a linear arrangement of actions) that can be generated between the initial state and final (goal) one. Agents can utilize the abductive EC planner to logically calculate protocol runs leading to a desirable outcome. By keeping track of agent's commitments, we can check whether the agent behaviors comply with its commitments. This technique can be named static verification that could be preferably carried out at run time. The technique seems promising, but computing all possible runs regarding the commitment in question is hard to apply when we check open systems having a large state space, as discussed in Artikis *et al.* (2009). Yolum and Singh also pointed out that their specifications achieve flexibility by means of enabling agents to adjust their actions by taking advantages of opportunities and accommodating exceptions that arise at run time by reconstructing plans as necessary.

However, the flexibility resulting from reasoning capabilities can be expensive and might increase the code of the agents (Yolum & Singh, 2002a; Singh, 2007). Thus, the authors suggested that the specification of protocols can be compiled into FSMs (Yolum & Singh, 2002a) or Büchi automata (Singh, 2007). To this end, the generated automata can be more complete to capture the important scenarios and in turn be too large for designers so as to specify, analyze, and verify manually.

To address the limitation of Yolum and Singh (2002b, 2004) regarding the verification issue, Yolum (2007) presented the main generic properties that can help protocol designers analyze and correct the development of commitment-based protocols by (1) signaling possible errors and inconsistencies; and (2) determining the applicability of protocols. Technically, such properties are categorized into three classes: *effectiveness*, *consistency,* and *robustness*. Yolum then presented algorithms that could be implemented in any 'available software tool' to semi-automatically verify those properties at design time. However, because there is no experimental results, then we cannot be able to evaluate the performance and efficiency of the introduced algorithms.

Based on Yolum and Singh's representation of commitment actions (Yolum & Singh, 2004) in terms of the EC axioms, Chesani *et al.* (2009) proposed a framework composed of a logical language and verification procedure. The language defines commitments, commitment actions, deadlines, and 'compensations actions' that can arise when deadline is expired. Technically, they modeled commitments as EC fluents and provided an axiomatization of commitment actions in terms of EC primitives. For example:

Example 28 *When a commitment C(Mer, Cus, deliverGoods) has been established and the deadline has not expired yet, a fluent waiting (C(Mer, Cus, deliverGoods)) holds:*

$$initiates(E, waiting(C(Mer, Cus, deliverGoods)), T) \leftarrow$$
$$create(E, Mer, C(Mer, Cus, deliverGoods))$$

where E is an event and T the absolute time at which the commitment has been established.

When an event regarding the discharge of this commitment occurs, the *waiting*(*C* (*Mer*, *Cus*, *deliverGoods*)) fluent is terminated and a new *satisfied* (*C* (*Mer*, *Cus*, *deliverGoods*)) fluent is instantiated, which in principle means the commitment has been successfully discharged. This would be expressed as:

$$terminates(E, waiting(C(Mer, Cus, deliverGoods)), T) \leftarrow$$
$$holds\_at(waiting(C(Mer, Cus, deliverGoods)), T),$$
$$discharge(E, Mer, C(Mer, Cus, deliverGoods))$$
$$initiates(E, Satisfied(C(Mer, Cus, deliverGoods)), T) \leftarrow$$
$$holds\_at(waiting(C(Mer, Cus, deliverGoods)), T),$$
$$discharge(E, Mer, C(Mer, Cus, deliverGoods))$$

A fluent *d_check* (C, T_D) is introduced to check whether a commitment C is satisfied at time T_D. It can be instantiated as:

$$initiates(E, d\_check(C(Mer, Cus, deliverGoods), When), T) \leftarrow$$
$$create(E, Mer, C(Mer, Cus, deliverGoods))$$
$$deadlines(C(Mer, Cus, deliverGoods), Delay)$$
$$When\ is\ T + Delay.$$

deadlines (*C* (*Mer*, *Cus*, *deliverGoods*), *3*) means that the commitment should be fulfilled within 3 time units from its instantiation. Their verification procedure (or monitoring tool) is developed to track the commitment statuses at run time. Specifically, they implemented the above defined language using REC. REC basically builds on top of the computational logic framework, named SCIFF (Alberti *et al.*, 2008), which in turn extends Fung and Kowalski's IFF proof procedure for abductive logic programming. Notice that the implementation of EC is performed using the jREC Java archive tool. The authors opted REC because it is 'not goal-directed but event-driven', that is the fluents' statuses can be modified at run time

when certain events occur. By defining the set of possibly observed events at certain times, the values of deadline and the corresponding fluents, the monitor tool computes a set of observed events that comply with the defined axioms encoding the set of possible events and then returns either true or the state of commitments in questions.

The authors (Chesani *et al.*, 2013) recently extended their CML Torroni *et al.*, (2010) by considering (1) conditional commitments; (2) data, variables, and metric time to deal with temporal aspects (e.g. deadlines) that are compatible with dynamic settings, but are missing in temporal logics (e.g. LTL, CTL, and CTL*); and (3) other commitment actions, such as *detach, expire*, and *violate*. The authors also mapped each possible commitment state into a fluent state to explicitly refer to 'the state in the domain dependent theory'; instead of setting a correspondence between 'bringing about some property *P*' and 'initiating fluent *P*' as done by Yolum and Singh (2004).

Example 29 *Consider the following axiom:*

$$create(promise(Ag_1, Ag_2, deliverGoods), C(Ag_1, Ag_2, property(e(T_1, T_2), deliverGoods)), T) \leftarrow T_1 \, is \, T+1, \, T_2 \, is \, T+3$$

Suppose we observe the following event: promise (*Mer*, *Cus*, *deliverGoods*) *at time 20. As this event copes with the description of create (…), then Mer becomes committed to deliver the requested goods between time 21 and time 23: C* (*Mer*, *Cus*, *property(e(21, 23)*, *deliverGoods*)).

From agent communication perspective, the proposed semantics for ACL messages meets the first four criteria. On the one hand, the main limitation of the proposed CML is that we cannot express commitments where antecedent and consequence 'overlap'. This is because a conditional commitment's consequence should be brought about strictly after the original commitment is detached, and in turn create a new one. Simply put, the consequence and antecedent of commitments should be held in different states. On the other hand, to check REC's scalability, the authors considered up to 20 000 rental cars (reflecting the state space) and up to 200 customers. In general, for a specification to be effective, and thus to reach its aim, it is fundamental to consider the characteristics of the interacting agents (e.g. local states, local actions, local polices, etc.) and of the formal modeled systems it will be settled in. Thus, we do not know how efficient such characteristics will be, if implemented, however, we suspect scalability might be an issue. This framework also lacks formal semantics for commitments because they are modeled as fluents (cf. Remark 1). As REC is in fact a first-order logic with variables and data, it is undecidable (Emerson, 1990). Thus, it would be hard to verify commitment-based protocol properties (e.g. reachability, safety, liveness) typically checked using model checking techniques (Bentahar *et al.*, 2012; El-Menshawy *et al.*, 2013b). By contrast, model checking appears to be less feasible, but it is not completely impossible if a real-time formalization is required (cf. real-time CTL supported by the NuSMV tool).

7.2 Action logical language: C+

Chopra and Singh (2006) specified commitment-based protocols using 'non-monotonic commitment machines' presented first in Chopra and Singh (2004) with the action logical description language C^+ developed in Giunchiglia *et al.* (2004). A non-monotonic commitment machine is a modification of a commitment machine (Yolum & Singh, 2002a) to consider situations when agents must 'act with incomplete information'; so they need non-monotonic or defeasible reasoning. The authors adopted C^+ as it is easy to add or remove certain interactions to an existing specification. As in EC, fluents and actions are the basic components in the C^+ language. In this approach, a protocol specification consists of a set of causal laws, which in turn link messages specified as actions that occur at particular time points to their effects on commitments. More precisely, description of actions specifies a transition system of a protocol, a graph with states and actions wherein commitments are modeled as 'inertial fluents' (cf. Remark 1 about the problem of using fluents to model commitments). The interpretation (i.e. semantic value) of such fluents persists from one state to the next state 'unless changed by some other law'. The authors presented a

Table 6 Summary of logical languages of actions and commitment-based agent communication

	Agent communication semantics				Commitment		
	For.	Dec.	Mea.	Ver.	Mod.	Sem.	Verification method
Yolum and Singh (2002b, 2004)	√	√	√	√	Predicate		Abductive event calculus planner
Winikoff (2005, 2007)	√	√	√	√	Predicate		
Chesani *et al.* (2009, 2013)	√	√	√	√	Fluent		SCIFF proof procedure
Chopra and Singh (2006)	√	√	√	√	Fluent		Static verification
Desai and Singh (2007)	√	√	√		Fluent		

set of rules in the form of axioms to represent and in turn define the intended meanings of commitment actions as follows (Chopra & Singh, 2004):

(1) $Create(i,j,p)$ *causes* $C(i,j,p)$ (2) $Discharge(i,j,p)$ *causes* $\neg C(i,j,p)$
(3) $Cancel(i,j,p)$ *causes* $\neg C(i,j,p)$ (4) $Delegate(i,j,p,k)$ *causes* $\neg C(i,j,p)$ & (k,j,p)
(5) $Release(i,j,p)$ *causes* $\neg C(i,j,p)$ (6) $Assign(i,j,p,k)$ *causes* $\neg C(i,k,p)$ & (i,j,p)

For instance, the axiom of the discharge action means that when the discharge action happens, then the commitment is terminated ($\neg C(i,j,p)$) in 'the next state' according to the following rule (Chopra & Singh, 2004): *a causes f*, where *a* is an action and *f* is a fluent happening in the next state. Notably, this approach meets formal, declarative, and meaningful criteria. As in Yolum and Singh (2004), the authors exploited a static verification technique in the form of reasoning rules in order to verify the compliance of agent behaviors against a given protocol's state machine. As mentioned in the evaluation of Yolum and Singh (2004), this technique is inapplicable in complex systems because the verification procedure needs to record all possible protocol runs from initial and final protocol states to search for commitment states in question.

Desai and Singh (2007) presented a 'modular action description' geared toward protocols, an extension of the causal logic C^+ (Giunchiglia *et al.*, 2004) in order to refine and compose protocols from existing ones. This approach enhances the approach of Chopra and Singh (2006) for representing individual protocols. However, as in Chopra and Singh (2006), Desai *et al.*'s commitments are modeled as inertial fluents and the operational semantics of commitment actions is delineated by a set of axioms. Composition rules satisfying some requirements are also defined as a set of C^+ axioms without checking the correctness specification of a protocol composition. Such a challenge is addressed in Desai *et al.* (2007) (cf. Section 4).

To conclude this section, the advantages and disadvantages of EC and C^+ and the comparisons between them are discussed in Artikis *et al.* (2009). Artikis and Pitt (2009) showed that the abductive EC planner and causal calculator that execute commitment-based protocols specified in EC and C^+ can be employed, 'in principle, for the provision of both design time services—for instance, proving protocol properties—and run-time services—for example, calculating the commitments current at each time—for the benefit of the agents or their designers. These implementations, however, can become inefficient when considering large OASs [open agent systems]'. On the other hand, model checking in principle provides a full automatic verification and is effective in large and complex systems. Furthermore, there is no formal semantics for commitments as they are modeled simply as fluents (cf. Remark 1). Table 6 summarizes our results of evaluating the proposals that advocate those logical languages of actions.

8 Summary and suggestions for future work

In this article, we reviewed and evaluated most prominent proposals that have advocated computational logics to define semantics for ACL messages in terms of social commitments and related concepts and to use the resulting logical models to derive specification languages of commitment-based protocols. We also investigated different verification techniques that have been proposed to verify these protocols and

classified them into run-time verification techniques and design time verification techniques. And the latter techniques are implemented using either informal reduction methods or formal reduction methods. By highlighting the commonalities, advantages, and disadvantages, we hoped that designers can make an informed decision when choosing to take the advantages of ACL semantics that, in principle, satisfy our seven crucial criteria. We also reviewed other logical languages of actions adopted to specify, model, and execute commitment-based protocols. The overall conclusion is summarized in the following points, which particularly state the limitations of those proposals from our perspective:

1. Modeling commitments and their actions as predicates, fluents, or domain variables waives formal semantics and concrete meaning of commitments, which in principle have opaque context.
2. Using informal reduction-based techniques to be able to use existing model checkers raises the following issues:
   - Such techniques have the problem of preventing verifying the real semantics of commitments and commitment actions as defined in the underlying logics.
   - We cannot prove their correctness as the relationship between the original and obtained models is not delineated by a formal way.
   - They lack a dedicated and well-adapted model checking algorithm.
3. The use of abductive EC planner and causal calculator implementations—which execute commitment-based protocols specified in the EC and C^+ languages—'for the provision of run time services may be limited' (Artikis & Pitt, 2009).

The idea of modeling commitments and associated actions as modal operators is in fact not new. In the literature about agent communication, few research proposals have supported such idea (Singh, 2000; Bentahar *et al.*, 2007; Singh, 2008; El-Menshawy *et al.*, 2009a, 2010b, 2011b, 2011c). On the one hand, although developing such modalities is far from being easy, it allows us to:

1. Combine the sense of formula along with its reference to qualify their truth values and to have a standard form for representing and reasoning about social notions having opaque contexts.
2. Define deemed appropriate axioms incorporated with those modal operators. Thus, the validity of those modal operators will be limited to models that only fulfill the constraints imposed by such axioms.
3. Develop dedicated and implementable model checking algorithms for commitments and their actions and thereby commitment-based protocols. Such algorithms address the limitations of formal reduction methods with regard to scalability and high memory usage. Recall that formal reduction methods are developed to tackle the shortcomings of informal reduction methods in terms of lacking both formal reduction rules and the possibility of proving soundness.

On the other hand, when the designers for some reason do not need temporal modalities, then they can model commitments as fluents, predicates, domain variables, and propositions to avoid the hardness of using temporal modalities and to be able to utilize run-time verification techniques. Those techniques, however, have some limitations with regard to scalability and the fact that they do not allow verifying protocols against desirable properties as in model checking. However, model checking only allows us to perform design time verification, not run-time one. Furthermore, although design time verification techniques take more execution time to accomplish, they are acceptable for detecting design errors for at least two crucial reasons. The former reason is about reducing the cost and time requirements needed for both implementation and maintenance phases. The latter reason is about avoiding catastrophic situations in terms of both human lives loss and economic damages (think of failing safety property in control systems of nuclear power plants). To summarize, model checking and run-time verification are not competing; they are rather complementing each other.

For those reasons, we suggest to develop a framework that combines design time verification techniques with run-time verification techniques to achieve the benefits of both techniques. Of course, this framework should meet our seven crucial criteria. Technically, designers can start with design time verification techniques to remove design errors before starting to track the behaviors of agents so as to ensure that the observed bad behaviors are not resulting from errors in the specifications. Unfortunately, the current design time verification techniques substantially suffer from lacking (1) a suitable semantics for

conditional commitment modalities, the universal frame of social commitments (the reasons are in our discussion about recent proposals (Singh, 2008; El-Menshawy *et al.*, 2013a)) and the model checking algorithm of this frame; (2) a suitable semantics for fulfillment modality (the reasons are in our discussion about recent proposals (El-Menshawy *et al.*, 2011c, 2013b; Bentahar *et al.*, 2012)); and (3) formal semantics for other commitment actions modalities applied directly to conditional commitments and their model checking algorithms.

To address the first limitation, we suggest to distinguish between two different but related types of conditional commitments: weak and strong. Weak conditional commitments are those that can be created even if the antecedent will never be satisfied, whereas strong conditional commitments are only created when there is a possibility to satisfy their antecedents (El-Kholy *et al.*, 2014a, 2015). Notice that as agents are able to decide which commitments are relevant for their goals and because in a model checking approach (Clarke *et al.*, 1999), the model should be finite, then it is easy for agent to check the satisfiability of its commitment antecedent. In addition, there are several reasonable and intuitive business scenarios, which require the satisfaction of antecedents; first, to consequently achieve some tasks of interest, such as a surgeon cannot strongly commit to replace an organ unless there is assuring possibility to find a good one (El-Kholy *et al.*, 2014a). By doing so, the pressing question is which logical language we should select? We suggest to use a language that balances between expressive powers and efficient verification complexity. From Tables 1, 2 and 4, we can observe that the number of proposals that utilize the computational logics LTL and CTL to define artificial languages that artificial agents can use in terms of social notions (commitments and commitment-based protocols) are more than those that use CTL*, although CTL* is more expressive than LTL and CTL. To answer this question, we suggest to utilize CTL for three reasons: (1) there are many open model checkers that support CTL model checking algorithm; (2) although the expressiveness of LTL and CTL is incomparable, the standard model checking algorithm for LTL and CTL* are exponential in the size of the formula and linear in the size of the model (Clarke *et al.*, 1999; Schnoebelen, 2003) and for CTL, the algorithm is linear in both the size of the formula and model (Clarke *et al.*, 1999; Schnoebelen, 2003); and (3) we proved in Bentahar *et al.* (2012) and El-Menshawy *et al.* (2013b) that a symbolic model checking algorithm of an extension of CTL with modalities for unconditional commitments and their fulfillments has the same computational complexity of standard CTL symbolic model checking algorithm. To address the second limitation, we can modify the semantics of fulfillment modality to terminate the active commitment in the fulfillment state (El-Kholy *et al.*, 2014a). The semantics of other commitment actions modalities and their model checking algorithms are still unsolved problems. For the run-time verification techniques that can be employed in this framework, they should have formal models—which are missing in the literature—in order to transform (reduce) logical models into them.

For other directions of future work, we suggest to analyze commitments and related actions using strategic logics such as ATL* (Alur *et al.*, 2002) where agents can commit to specific strategies. We also suggest to combine CTLK (an extension of CTL with knowledge modality (Penczek & Lomuscio, 2003)) and CTLC (Bentahar *et al.*, 2012; El-Menshawy *et al.*, 2013b) to study the relationship between agents knowledge and commitments. Al-Saqqar *et al.* (2014) studied this relationship in a logic called CTLKC+ along with a reduction technique (Al-Saqqar *et al.*, 2015) to verify the correctness of the CTLKC+ specifications. However, they are only focused on unconditional commitments and their fulfillment without considering (1) other unconditional commitment actions; and (2) conditional commitments and their actions. Moreover, we suggest to study the relationship between common and distributed knowledge and commitments that underwrite much of social life scenarios. Furthermore, the relationship between probability and unconditional commitments and their fulfillments has been studied in Sultan *et al.* (2013, 2014) in a logic called probabilistic computation tree logic of commitments (PCTLC). However, there is still a need to develop a model checking algorithm to directly verify the correctness of the PCTLC specifications and to study probabilistic conditional commitment and their actions. Recently, El-Kholy *et al.* (2014b, 2014c) used their logical language introduced in El-Kholy *et al.* (2014a) to express commitment protocol properties. Then, they introduced a formal language for commitment protocols, which regulate interactions among multi-agent-based Web services within the composition process in which interactions themselves are modeled by conditional commitments. However, this direction of

research still needs to investigate how to compose different commitment protocols to model complex business scenarios. Moreover, they abstracted conditional commitment actions as 'predicate propositions' to keep their logical language pure temporal logic and to avoid formally defining the formal semantics of these actions. In all of the suggested future work, we need to develop suitable model checking algorithms and then analyze their computational complexity (space and time).

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions of changes and improvements. The authors also would like to thank NSERC (Canada) and Menofia University (Egypt) for their financial support.

References

Alberti, M., Chesani, F., Gavanelli, M., Lamma, E., Mello, P. & Torroni, P. 2008. Verifiable agent interaction in abductive logic programming: the SCIFF framework. *ACM Transactions on Computational Logic* **9**(4), 1–43.

Al-Saqqar, F., Bentahar, J., Sultan, K. & El-Menshawy, M. 2014. On the interaction between knowledge and social commitments in multi-agent systems. *Applied Intelligence* **41**(1), 235–259.

Al-Saqqar, F., Bentahar, J., Sultan, K., Wan, W. & Khosrowshahi Asl, E. 2015. Model checking temporal knowledge and commitments in multi-agent systems using reduction. *Simulation Modelling Practice and Theory* **51**, 45–68.

Alur, R., Henzinger, T.A. & Kupferman, O. 2002. Alternating-time temporal logic. *Journal of ACM* **49**(5), 672–713.

Artikis, A. & Pitt, J.V. 2009. Specifying open agent systems: a survey. In *ESAW*, Artikis, A., Picard, G. & Vercouter, L. (eds), Lecture Notes in Computer Science **5485**, 29–45. Springer.

Artikis, A., Sergot, M. & Pitt, J. 2009. Specifying norm-governed computational societies. *ACM Transactions on Computational Logic* **10**(1), 1–42.

Baldoni, M., Baroglio, C. & Marengo, E. 2010. Behavior oriented commitment-based protocols. In *ECAI*, Coelho, H., Studer, R. & Wooldridge, M. (eds), **215**. IOS Press, 137–142.

Baldoni, M., Baroglio, C. & Marengo, E. 2011. Commitment-based protocols with behavioral rules and correctness properties of MAS. In *DALT*, Omicini, A., Sardiña, S. & Vasconcelos, W. (eds), LNCS **6619**. 60–77. Springer.

Baldoni, M., Baroglio, C., Marengo, E. & Patti, V. 2013. Constitutive and regulative specifications of commitment protocols: a decoupled approach. *ACM Transactions on Intelligent Systems and Technology* **4**(2), 22.

Bentahar, J., El-Menshawy, M., Qu, H. & Dssoulia, R. 2012. Communicative commitments: model checking and complexity analysis. *Knowledge-Based Systems* **35**, 21–34. Elsevier.

Bentahar, J., Maamar, Z., Wan, W., Benslimane, D., Thiran, P. & Subramanian, S. 2008. Agent-based communities of web services: an argumentation-driven approach. *Service Oriented Computing and Applications* **2**(4), 219–238.

Bentahar, J., Meyer, J.-J.Ch. & Wan, W. 2010. Model checking agent communication. In *Specification and Verification of Multi-Agent Systems*, chapter 3, 1st edition, Dastani, M., Hindriks, K. & Meyer, J.-J.Ch. (eds). Springer, pp. 67–102.

Bentahar, J., Meyer, J.-J.Ch. & Wan, W. 2009. Model checking communicative agent-based systems. *Knowledge-Based Systems* **22**, 142–159.

Bentahar, J., Moulin, B. & Chaib-draa, B. 2003. Towards a formal framework for conversational agents, In *Proceedings of the International Workshop on ACLCP*.

Bentahar, J., Moulin, B. & Chaib-draa, B. 2004a. Commitment and argument network: a new formalism for agent communication. In *ACL*, Dignum, F. (ed.), LNCS **2922**. 146–165. Springer.

Bentahar, J., Moulin, B., Meyer, J.-J.Ch. & Chaib-draa, B. 2004b. A logical model for commitment and argument network for agent communication. In *Proceedings of the 3rd International Conference on AAMAS*, 792–799. IEEE Computer Society.

Bentahar, J., Moulin, B., Meyer, J.-J.Ch. & Lespérance, Y. 2007. A new logical semantics for agent communication. In *CLIMA*, Inoue, K., Satoh, K. & Toni, F. (eds), LNCS **4371**. 151–170. Springer.

Bhat, G., Cleaveland, R. & Groce, A. 2001. Efficient model checking via Büchi tableau automata. In *CAV*, Berry, G., Comon, H. & Finkel, A. (eds), LNCS **2102**. 38–52. Springer.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. & Mylopoulos, J. 2004. Tropos: an agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems* **8**(3), 203–236.

Cabri, G., Leonardi, L., Ferrari, L. & Zambonelli, F. 2010. Role-based software agent interaction models: a survey. *Knowledge Engineering Review* **25**(4), 397–419.

Castelfranchi, C. 1995. Commitments: from individual intentions to groups and organizations. In *ICMAS*, Lesser, V. & Gasser, L. (eds). The MIT Press, 41–48.

Chesani, F., Mello, P., Montali, M. & Torroni, P. 2013. Representing and monitoring social commitments using the e. *Autonomous Agents and Multi-Agent Systems* **27**(1), 85–130.

Chesani, F., Mello, P., Montali, M. & Torroni, P. 2009. Commitment tracking via the reactive event calculus. In *IJCAI*, Boutilier, C. (ed.). AAAI Press, 91–96.

Chopra, A. & Singh, M. 2004. Nonmonotonic commitment machines. In *ACL*, Dignum, F. (ed.), LNCS **2922**. 183–200. Springer.

Chopra, A. & Singh, M. 2006. Contextualizing commitment protocols. In *AAMAS*, Nakashima H., Wellman M., Weiss, G. & Stone, P. (eds). ACM, 1345–1352.

Chopra, A. & Singh, M. 2008. Constitutive interoperability. In *AAMAS*, Padgham, L., Parkes, D., Müller, J. & Parsons, S. (eds), **2**. IFAAMAS, 797–804.

Chopra, A. & Singh, M. 2009. Multiagent commitment alignment. In *Proceedings of the 8th International Joint Conference on AAMAS*, 937–944. ACM Press.

Chopra, A.K., Artikis, A., Bentahar, J., Colombetti, M., Dignum, F., Fornara, N., Jones, A.J.I., Singh, M.P. & Yolum, P. 2013. Research directions in agent communication. *ACM Transactions on Intelligent Systems and Technology* **4**(2), 20. ACM.

Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R. & Tacchella, A. 2002. NuSMV: an open source tool for symbolic model checking. In *CAV*, Brinksma, E. & Larsen, K.G. (eds), LNCS **2404**. 359–364. Springer.

Clarke, E. & Emerson, E. 1982. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Logics of Programs*, Kozen, D. (ed.), LNCS **131**. 52–71.

Clarke, E., Emerson, E. & Sistla, A. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. In *Proceedings of the 10th ACM SIGACT-SIGPLAN Symposium on PPL*, POPL'83, 117–126. ACM.

Clarke, E., Grumberg, O. & Peled, D. 1999. *Model Checking*. The MIT Press, Cambridge, USA.

Cohen, P. & Levesque, H. 1990. Intention is choice with commitment. *Artificial Intelligence* **42**(2–3), 213–261.

Colombetti, M. 2000. A commitment-based approach to agent speech acts and conversations. In *Proceedings of International Workshop on ALCP, 4th International Conference on Autonomous Agents (Agents 2000)*, 21–29.

Colombetti, M., Fornara, N. & Verdicchio, M. 2004. A social approach to communication in multiagent systems. In *DALT*, Leite, J.A., Omicini, A., Sterling, L. & Torroni, P. (eds), LNCS **2990**. 191–220. Springer.

Desai, N., Cheng, Z., Chopra, A. & Singh, M. 2007. Toward verification of commitment protocols and their compositions. In *AAMAS*, Durfee, E.H., Yokoo, M., Huhns, M.N. & Shehory, O. (eds). IFAAMAS, 144–146.

Desai, N., Mallya, A., Chopra, A. & Singh, M. 2005. Interaction protocols as design abstractions for business processes. *IEEE Transactions on Software Engineering* **31**(12), 1015–1027.

Desai, N. & Singh, M. 2007. A modular action description language for protocol composition. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, 962–967.

Dignum, F. & Greaves, M. (eds) 2000. *Issues in Agent Communication*, LNCS **1916**. Springer.

El-Kholy, W., Bentahar, J., El-Menshawy, M., Qu, H. & Dssouli, R. 2014a. Conditional commitments: reasoning and model checking. *ACM Transaction on Software Engineering and Methodology* **24**(2), 9:1–9:49. ACM.

El-Kholy, W., Bentahar, J., El-Menshawy, M., Qu, H. & Dssouli, R. 2014b. Modeling and verifying choreographed multi-agent-based web service compositions regulated by commitment protocols. *Expert Systems with Applications* **41**, 7478–7494. Elsevier.

El-Kholy, W., El-Menshawy, M., Bentahar, J., Qu, H. & Dssouli, R. 2014c. Verifying multiagent-based web service compositions regulated by commitment protocols. In *Proceedings of 21th IEEE International Conference on Web Services (ICWS)*, 49–56. IEEE.

El-Kholy, W., El-Menshawy, M., Bentahar, J., Qu, H. & Dssouli, R. 2015. Formal specification and automatic verification of conditional commitments. *IEEE Intelligent Systems* **30**(2), 36–44. IEEE.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2009a. Enhancing engineering methodology for communities of web services. In *MALLOW*, Baldoni, M. *et al*. (eds), **494**. CEUR-WS.org. pp. 33–42.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2009b. An integrated semantics of social commitments and associated operations. In *MALLOW*, Baldoni, M. *et al.* (eds), **494**. CEUR-WS.org. pp. 222–230.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2010a. Modeling and verifying business interactions via commitments and dialogue actions. In *KES-AMSTA*, Jedrzejowicz, P., Nguyen, N.T., Howlett, R.J. & Jain, L.C. (eds), LNCS **6071**. 11–21. Springer.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2010b. Verifiable semantic model for agent interactions using social commitments. In *LADS*, Dastani, M., Fallah-Seghrouchni, A.E., Leite, J. & Torroni, P. (eds), LNCS **6039**. 128–152. Springer.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2011a. Model checking commitment protocols. In *IEA–AIE*, Mehrotra, K.G. *et al*. (eds), LNCS **6704**. 37–47. Springer.

El-Menshawy, M., Bentahar, J. & Dssouli, R. 2011b. Symbolic model checking commitment protocols using reduction. In *DALT*, Omicini, A., Sardina, S. & Vasconcelos, W. (eds), LNAI **6619**. 185–203. Springer.

El-Menshawy, M., Bentahar, J., El-Kholy, W. & Dssouli, R. 2013a. Verifying conformance of multi-agent commitment-based protocols. *Expert Systems with Applications* **40**(1), 122–138. Elsevier.

El-Menshawy, M., Bentahar, J., El-Kholy, W. & Dssouli, R. 2013b. Reducing model checking commitments for agent communication to model checking ARCTL and GCTL*. *Autonomous Agent Multi-Agent Systems* **27**(3), 375–418. Springer.

El-Menshawy, M., Bentahar, J., Qu, H. & Dssouli, R. 2011c. On the verification of social commitments and time. In *AAMAS*, Sonenberg, L., Stone, P., Tumer, K. & Yolum, P. (eds). IFAAMAS, 483–490.

Emerson, E. 1990. Temporal and modal logic. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics (B)*, chapter 16, van Leeuwen, J. (ed.). Elsevier, 995–1072.

Emerson, E. & Halpern, J. 1986. Sometimes and not never, revisited: on branching versus linear time temporal logic. *Journal of ACM* **33**(1), 151–178.

Fagin, R., Halpern, J., Moses, Y. & Vardi, M. 1995. *Reasoning About Knowledge*. The MIT Press.

Fornara, N. & Colombetti, M. 2002. Operational specification of a commitment-based agent communication language. In *Proceedings of the 1st International Joint Conference on AAMS*, 535–542. ACM.

Fornara, N. & Colombetti, M. 2003. Defining interaction protocols using a commitment-based agent communication language. In *Proceedings of the 2nd International Joint Conference on AAMAS*, 520–527. ACM.

Fornara, N., Viganò, F., Verdicchio, M. & Colombetti, M. 2008. Artificial institutions: a model of institutional reality for open multi-agent systems. *AI and Law* **16**(1), 89–105.

Gascueña, J. & Fernández-Caballero, A. 2011. On the use of agent technology in intelligent, multisensory and distributed surveillance. *Knowledge Engineering Review* **26**(2), 191–208.

Gerard, S. & Singh, M. 2013. Formalizing and verifying protocol refinements. *ACM Transactions on Intelligent Systems and Technology* **4**(2), 21.

Giordano, L., Martelli, A. & Schwind, C. 2003. Specifying and verifying systems of communicating agents in a temporal action logic. In *AI*IA*, Cappelli, A. & Turini, F. (eds), LNCS **2829**. 262–274. Springer.

Giordano, L., Martelli, A. & Schwind, C. 2007. Specifying and verifying interaction protocols in a temporal action logic. *Journal of Applied Logic* **5**(2), 214–234.

Giunchiglia, E., Lee, J., Lifschitz, V., McCain, N. & Turner, H. 2004. Nonmonotonic causal theories. *Artificial Intelligence* **153**(1–2), 49–104.

Gray, J. 1978. Notes on database operating systems. In *Advanced Course: Operating Systems*, Flynn, M.J., Gray, J., Jones, A.K., Lagally, K., Opderbeck, H., Popek, G.J., Randell, B., Saltzer, J.H. & Wiehle, H. (eds), LNCS **60**. 393–481.

Günay, A. & Yolum, P. 2013. Constraint satisfaction as a tool for modeling and checking feasibility of multiagent commitments. *Applied Intelligence* **39**(3), 489–509.

Habermas, J. 1984. *The Theory of Communicative Action*. The Polity Press.

Hamblin, C. 1970. *Fallacies*. Methuen.

Holzmann, G. 1997. The model checker SPIN. *Software Engineering* **23**(5), 279–295.

Huth, M. & Ryan, M. 2004. *Logic in Computer Science: Modelling and Reasoning About System*, 2nd edition. Cambridge University Press.

Jennings, N. 1993. Commitments and conventions: the foundation of coordination in multi-agent systems. *Knowledge Engineering Review* **8**(3), 223–250.

Kafali, Ö., Günay, A. & Yolum, P. 2014. Detecting and predicting privacy violations in online social networks. *Distributed and Parallel Databases* **32**, 161–190.

Kafali, Ö. & Torroni, P. 2011. Social commitment delegation and monitoring. In *CLIMA XII*, Leite, J., Torroni, P., Ågotnes, T., Boella, G. & van der Torre, L. (eds), Lecture Notes in Computer Science **6814**. 171–189. Springer.

Kafali, Ö. & Torroni, P. 2012. Exception diagnosis in multiagent contract executions. *Annals of Mathematics and Artificial Intelligence* **64**(1), 73–107.

Labrou, Y. & Finin, T. 1998. Semantics and conversations for an agent communication language. In *ATAL*, Singh, M. P., Rao, A.S. & Wooldridge, M. (eds), LNCS **1365**. 209–214. Springer.

Lim, M. & Zhang, Z. 2012. A multi-agent system using iterative bidding mechanism to enhance manufacturing agility. *Expert Systems with Applications* **39**, 8259–8273.

Lomuscio, A., Pecheur, C. & Raimondi, F. 2007. Automatic verification of knowledge and time with NuSMV. In *Proceedings of the 20th International Joint Conference on AI*, 1384–1389.

Lomuscio, A., Qu, H. & Raimondi, F. 2009. MCMAS: a model checker for the verification of multi-agent systems. In *CAV*, Bouajjani, A. & Maler, O. (eds), LNCS **5643**. 682–688. Springer.

Lomuscio, A., Qu, H. & Solanki, M. 2012. Towards verifying contract regulated service composition. *Autonomous Agents and Multi-Agent Systems* **24**(3), 345–373.

Mallya, A. & Huhns, M. 2003. Commitments among agents. *IEEE Internet Computing* **7**(4), 90–93.

Mallya, A. & Singh, M. 2007. An algebra for commitment protocols. *Autonomous Agents and Multi-Agent Systems* **14**(2), 143–163.

Mallya, A., Yolum, P. & Singh, M. 2004. Resolving commitments among autonomous agents. In *ACL*, Dignum, F. (ed.), LNCS **2922**. 166–182. Springer.

Marengo, E., Baldoni, M., Baroglio, C., Chopra, A., Patti, V. & Singh, M. 2011. Commitments with regulations: reasoning about safety and control in REGULA. In *AAMAS*, Tumer, K., Yolum, P., Sonenberg, L. & Stone, P. (eds). IFAAMAS, 467–474.

Maudet, N. & Chaib-Draa, B. 2002. Commitment-based and dialogue-game based protocols: new trends in agent communication languages. *Knowledge Engineering Review* **17**(2), 157–179.

Nakano, M., Hasegawa, Y., Funakoshi, K., Takeuchi, J., Torii, T., Nakadai, K., Kanda, N., Komatani, K., Okuno, H. & Tsujino, H. 2011. A multi-expert model for dialogue and behavior control of conversational robots and agents. *Knowledge-Based Systems* **24**(2), 248–256.

Pecheur, C. & Raimondi, F. 2007. Symbolic model checking of logics with actions. In *Model Checking and Artificial Intelligence*, Edelkamp, S. & Lomuscio, A. (eds), LNCS **4428**. 113–128. Springer.

Penczek, W. & Lomuscio, A. 2003. Verifying epistemic properties of multi-agent systems via bounded model checking. *Fundamenta Informaticae* **55**(2), 167–185.

Pham, D. & Harland, J. 2007. Temporal linear logics as a basis for flexible agent interactions. In *AAMAS*, Durfee, E., Yokoo, M., Huhns, M. & Shehory, O. (eds). 124–131.

Pnueli, A. 1977. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on FOCS*, 46–57. IEEE Computer Society Press.

Reichenbach, H. 1947. *Elements of Symbolic Logic*. Macmillan.

Richiardia, M. 2012. Agent-based computational economics: a short introduction. *Knowledge Engineering Review* **27**(2), 137–149.

Sacerdoti, E. 1977. *The Structure of Plans and Behavior*. Elsevier.

Schnoebelen, P. 2003. The complexity of temporal logic model checking. In *Advances in Modal Logic*, Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter & Michael Zakharyaschev (eds), Advances in Modal Logic 4. 1–44. King's College Publications.

Searle, J. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

Shanahan, M. 1997. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*. The MIT Press.

Shanahan, M. 2000. An abductive event calculus planner. *Logic Programming* **44**, 207–239.

Singh, M. 1991. Social and psychological commitments in multiagent systems. In *AAAI Fall Symposium on KA at SOL*, 104–106.

Singh, M. 1996. *A Conceptual Analysis of Commitments in Multiagent Systems*. Technical report, North Carolina State University.

Singh, M. 1997. Commitments among autonomous agents in information-rich Environments. In *MAAMAW*, Boman, M. & van de Velde, W. (eds), LNCS **1237**. 141–155. Springer.

Singh, M. 1998. Agent communication languages: rethinking the principles. *IEEE Computer Society* **31**(12), 40–47.

Singh, M. 1999. An ontology for commitments in multiagent systems: toward a unification of normative concepts. *AI and Law* **7**(1), 97–113.

Singh, M. 2000. A social semantics for agent communication languages. In *Issues in Agent Communication*, Dignum, F. & Greaves, M. (eds), LNCS **1916**. 31–45. Springer.

Singh, M. 2007. Formalizing communication protocols for multiagent systems. In *IJCAI*, Veloso, M.M. (ed.). Morgan Kaufmann Publishers Inc., 1519–1524.

Singh, M. 2008. Semantical considerations on dialectical and practical commitments. In *AAAI*, Fox, D. & Gomes, C.P. (eds). AAAI Press, 176–181.

Singh, M. & Chopra, A. 2010. Programming multiagent systems without programming agents. In *ProMAS*, Braubach, L., Briot, J.-P. & Thangarajah, J. (eds), LNCS **5919**. 1–14. Springer.

Singh, M., Chopra, A. & Desai, N. 2009. Commitment-based service-oriented architecture. *IEEE Computer* **42**(11), 72–79.

Sirbu, M. 1997. Credits and debits on the internet. *IEEE Spectrum* **34**(2), 23–29.

Sklar, E. & Richards, D. 2010. Agent-based systems for human learners. *Knowledge Engineering Review* **25**(2), 111–135.

Spoletini, P. 2005. *Verification of Temporal Logic Specification via Model Checking*. PhD thesis, Politecnico di Milano.

Spoletini, P. & Verdicchio, M. 2007. Commitment monitoring in a multi-agent system. In *CEEMAS*, Burkhard, H.-D., Lindemann, G., Verbrugge, R. & Varga, L.Z. (eds), LNCS **4696**. 83–92. Springer.

Spoletini, P. & Verdicchio, M. 2009. An automata-based monitoring technique for commitment-based multiagent systems. In *COIN*, Hübner, J.F., Matson, E.T., Boissier, O. & Dignum, V. (eds), LNCS **5428**. 172–187. Springer.

Sultan, K., Bentahar, J. & El-Menshawy, M. 2014. Model checking probabilistic social commitments for intelligent agent communication. *Applied Soft Computing* **22**, 397–409.

Sultan, K., El-Menshawy, M. & Bentahar, J. 2013. Reasoning about social commitments in the presence of uncertainty. In *IEEE 12th International Conference on Intelligent Software Methodologies, Tools and Techniques*, 29–35.

Telang, P. & Singh, M. 2009a. Business modeling via commitments. In *SOCASE*, Kowalczyk, R., Vo, Q., Maamar, Z. & Huhns, M. (eds), LNCS **5907**. 111–125. Springer.

Telang, P. & Singh, M. 2009b. Enhancing Tropos with commitments. In *Conceptual Modeling: Foundations and Applications*, Borgida, A., Chaudhri, V.K., Giorgini, P. & Yu, E.S.K. (eds), LNCS **5600**. 417–435. Springer.

Telang, P. & Singh, M. 2012. Specifying and verifying cross-organizational business models: an agent-oriented approach. *IEEE Transactions on Services Computing* **5**(3), 305–318. IEEE.

Torroni, P., Chesani, F., Mello, P. & Montali, M. 2010. Social commitments in time: satisfied or compensated. In *DALT*, Baldoni, M., Bentahar, J., van Riemsdijk, M.B. & Lloyd, J. (eds), LNCS **5948**. 228–243. Springer.

Venkatraman, M. & Singh, M. 1999. Verifying compliance with commitment protocols: enabling open web-based multiagent systems. *Autonomous Agents and Multi-Agent Systems* **2**(3), 217–236.

Verdicchio, M. & Colombetti, M. 2003. A logical model of social commitment for agent communication. In *Proceedings of the 2nd International Joint Conference on AAMAS*, 528–535.

Verdicchio, M. & Colombetti, M. 2004. A logical model of social commitment for agent communication. In *ACL*, Dignum, F. (ed.), LNCS **2922**. 128–145. Springer.

Verdicchio, M. & Colombetti, M. 2005. Dealing with time in content language expressions. In *AC*, van Eijk, R., Huget, M. & Dignum, F. (eds), LNCS **3396**. 91–105. Springer.

Verdicchio, M. & Colombetti, M. 2006. From message exchanges to communicative acts to commitments. *Electronic Notes in Theoretical Computer Science* **157**, 75–94.

Vidoni, R., Garca-Sánchez, F., Gasparetto, A. & Martnez-Béjar, R. 2011. An intelligent framework to manage robotic autonomous agents. *Expert Systems with Applications* **38**, 7430–7439.

Walton, D. & Krabbe, E. 1995. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press.

Weiss, G. 1999. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. The MIT Press.

Winikoff, M. 2007. Implementing commitment-based interactions. In *AAMAS*, Durfee, E., Yokoo, M., Huhns, M. & Shehory, O. (eds). ACM, 873–880.

Winikoff, M., Liu, W. & Harland, J. 2005. Enhancing commitment machines. In *DALT*, Leite, J.A., Omicini, A., Torroni, P. & Yolum, P. (eds), LNCS **3476**. 198–220. Springer.

Wooldridge, M. 2000. Semantic issues in the verification of agent communication languages. *Autonomous Agents and Multi-Agent Systems* **3**(1), 9–31.

Wooldridge, M. 2009. *An Introduction to Multi-Agent Systems*. John Wiley and Sons.

Xing, J. & Singh, M. 2001. Formalization of commitment-based agent interaction. In *SAC*, 115–120. ACM.

Xing, J. & Singh, M. 2003. Engineering commitment-based multiagent systems: a temporal logic approach. In *Proceedings of the 2nd International Joint Conference on AAMAS*, 891–898.

Yolum, P. 2007. Design time analysis of multiagent protocols. *Data and Knowledge Engineering* **63**(1), 137–154.

Yolum, P. & Singh, M. 2002a. Commitment machines. In *ATAL*, Meyer, J.-J.Ch. & Tambe, M. (eds), LNCS **2333**. 235–247. Springer.

Yolum, P. & Singh, M. 2002b. Flexible protocol specification and execution: applying event calculus planning using commitments. In *Proceedings of the International Joint Conference on AAMAS*, 527–534. ACM.

Yolum, P. & Singh, M. 2004. Reasoning about commitments in the event calculus: an approach for specifying and executing protocols. *Annals of Mathematics and Artificial Intelligence* **42**(1–3), 227–253.