Commitment-based Protocols with Behavioral Rules and Correctness Properties of MAS

Matteo Baldoni, Cristina Baroglio, and Elisa Marengo

Dipartimento di Informatica — Università degli Studi di Torino
c.so Svizzera 185, I-10149 Torino (Italy)
{baldoni,baroglio,emarengo}@di.unito.it

Abstract. Commitment-based interaction protocols are a flexible way of representing the interaction of a set of agents, that is well-known and widely accepted by the research community. Normally these protocols consist of sets of actions with a shared meaning. From the point of view of an agent, however, the meaning of an action is completed by the context in which it is used: the context shapes the behavior of the agent in that the agent decides which actions to take depending on it. Indeed, since the seminal work of Searle (supported by other authors), two components of interaction protocols have been identified, constitutive rules and regulative rules, which altogether define the meaning of the interaction. Commitment-based protocols usually do not account for the latter. In this work we introduce a representation that explicitly includes regulative rules as constraints on commitments and, in the light of the work by Singh and Chopra [36], report the *first steps* in the analysis of the advantages brought by such introduction.

1 Introduction

The term "interaction protocol" refers to a pattern of behavior that allows a set of agents to become a multi-agent system when engaging the expected interactions with one another. Protocols can be seen as public artifacts [36], ruling the interaction of agents playing the various roles. A role specification is just a formal definition of what is lawful for its player to do or to expect at any possible state of the interaction. This specification is given independently from the player that will enact the role.

Considering protocols as *models of the desired interaction* allows one to devise the verification of many properties and guarantee them before any interaction takes place. For instance, it is possible to check if the roles of a protocol are interoperable, i.e. if they allow any interaction to take place. An agent which accepts to conform to a protocol, whose roles are proved interoperable, is ideally guaranteed that its interaction with any other agents, playing the other roles foreseen by the same protocol, will succeed [4, 30, 10]. This is surely an advantage [6] w.r.t. checking directly the interoperability and the properties of interaction of a set of agents: in this latter case, the verification of properties can only be done after the composition is made, against the system as a whole; thanks to

protocols, instead, the verification of the interoperability can be *distributed* in time and among the various agents that could take on the roles. A candidate role player could autonomously check its conformance to the model by comparing its behavior to the role that it means to play. To do this the agent does not need to have the implementations of the other roles. This modularity of the verification meets the requirements given by interaction protocol engineering.

Interaction protocols can be specified in different ways. Some representations, like proposals based on Petri nets, finite state machines or on Pi calculus have an algorithmic (procedural) nature that is suitable to capture the desired interaction flows. Singh and colleagues criticize the use of this kind of specification as being too rigid [14, 34, 42, 41]: agents cannot, for instance, take advantage of opportunities that arise along the interaction and that are not explicitly included in their procedure. These authors propose the more flexible *commitment-based protocols*. A commitment can be seen as a fluent which can hold in the social state of the system. It represents the fact that a debtor commits to a creditor to bring about some condition. All the agents that interact according to a commitment-based protocol share the semantics of a set of actions, which affect the social state by creating new commitments, canceling commitments, and so forth. The greatest advantages of the commitment-based protocols, w.r.t. other approaches to interaction, are that they *do not over-constrain* the behavior of the agents by imposing an ordering on the execution of the shared actions, and that by giving a shared meaning to the social actions, they allow working on actual knowledge of what happened (or what is likely to happen), rather than on beliefs about each others' mental state.

The only constraint that commitment-based protocols include, to specify that an interaction is successful, is that all commitments are discharged. The research question that we face in this work is whether the specification of sequences of actions as part of a protocol compromises the autonomy of agents or whether it is an instrument that gives additional meaning to actions, a meaning that we lose when we remove all constraints. As Cherry observes when commenting on Searle's work [32, 11], actions acquire meaning depending on the context they are used: for instance, *shaking hands* means "agreement reached" in a negotiation protocol. A commitment-based protocol can specify that the meaning of shaking hands is that the agreement was reached, however, it puts no constraint on when it makes sense to use the action. What if a person shakes hands with someone he/she would like to reach an agreement with *before* starting the negotiation? Something is missing. Sometimes authors fill this gap by enriching actions with preconditions to their (non-) executability [39, 16], in this way they rule the order of action execution.

In our view, an interaction protocol must not only specify the agreed meaning of actions but it must express also an agreement on the way the agents will behave and use the protocol actions. This should be done in a way that does not compromise the autonomy of agents, which would be free to decide how to act and to take advantage of opportunities, that arise along the interaction, taking also the risk of being misunderstood when they get out of the boundaries given

by the protocol. After an agreement we can shake hands twice, if we are happy to do so, but shaking hands before the agreement is not understandable in the context of that protocol.

In this paper, we take on the commitment-based interaction protocol model proposed in [7]. The main characteristic of this model is a decoupled representation of the constitutive and the regulative specifications of the protocol, which are both based on commitments. While the *constitutive* specification defines the meaning of actions based on their effects on the social state, the *regulative* specification is a set of behavioral rules, given in terms of constraints among commitments, which regulate the evolution of the social state independently from the executed actions. To the best of our knowledge, this decoupling, postulated since the seminal work of Searle [32, 11], was not implemented in commitment-based interaction protocols before [7]. Then we survey the properties hoped for interaction protocols in [36] and report some initial considerations about how the introduction of the regulative specification not only does not compromise the advantages, given by the commitment-based approach, in their verification but it also allows the verification of such properties in a finer and modular way because the specification of protocols meets the specification of agents.

2 Commitment-based Protocols

Commitment protocols [34, 41, 42] are interaction patterns given in terms of commitments, involving a set of predefined roles. Commitments are directed from a debtor to a creditor. The notation $C(x, y, r, p)$ denotes that the agent playing the role x commits to an agent playing the role y to bring about the condition p when the condition r holds. All commitments are conditional. An unconditional commitment is merely a special case where r equals *true*. Whenever this is the case, we use the short notation $C(x, y, p)$. Agents share a social state that contains commitments and other fluents that are relevant to their interaction. Every agent can affect the social state by executing actions, whose definition is given in terms of modifications to the social state (e.g. adding a new commitment, releasing another agent from some commitment, satisfying a commitment, etc.). So a commitment protocol is made of a *set of actions*, involving the foreseen roles and whose semantics is agreed upon by all of the participants [41, 42, 15].

On the other hand, agents show a *behavior*, which is not captured by the action definitions but it rather involves a decision process (a procedure, a goal-driven plan [38], etc.) aimed at selecting the action to execute [40, 31]. An autonomous agent situated in an environment *decides* which actions to perform depending on the particular situation it is facing. From a certain point of view, as discussed in the introduction, the meaning of an action is completed by the context in which it is used, e.g. after which other actions it is used.

Since protocols are intended to rule the interaction of agents, the expectation is that they show the same structure of agents. Indeed, Searle [32] and later other authors, e.g. [11, 9, 17], have pointed out the need for a distinction between the *regulative* and the *constitutive* specifications of an interaction protocol. The

constitutive specification gives the semantics of actions, while the regulative one rules the *flow of execution*. The regulative specification, encoding the behavioral rules, however, is not explicitly represented in commitment-based approaches like [17, 15, 36, 39, 23, 41], where only actions are represented.

An actual identification, not only in agents but also *in the protocol definition itself*, of two separated components (the constitutive specification and the regulative specification) we argue would bring many advantages in the construction of multi-agent systems. The *decoupling* of the two parts would allow an *easier re-use* of actions in different contexts, an *easier customization* on the protocol, an *easier composition* of protocols. As a consequence, multi-agent systems would gain greater *openness*, *interoperability*, and *modularity* of design. In particular, interoperability would be better supported because it would be possible to verify it w.r.t. specific aspects (e.g. interoperability at the level of actions [17, 15, 18] or at the level of regulation rules). Protocols would be more open in the sense that their modularity would allow designers to easily adapt them to different contexts. Moreover, it would be possible to check properties that concern a single agent, willing to play a role of the protocol, *against the protocol and independently from which other agents will play the other roles*. In other words, if an agent in a system is substituted by another agent, it would not be necessary to recheck the whole system from scratch, because certain verifications can be distributed.

In the literature it is possible to find approaches that include in the protocol representation some regulative specification. For instance [35], where *before* relations are applied to events to define rules of behavior, like [26], where preferences about alternative behaviors are specified, like [3], where temporal constraints among the times at which events occur are specified, or like [22], where interaction diagrams are introduced inside protocols to rule the use of actions.

Unfortunately, even when behavioral rules are explicitly represented in some way, the decoupling between the regulative and the constitutive specification is not sufficiently supported yet, see [7] for details. Our proposal, which is described hereafter, explicitly accounts for decoupled constitutive and regulative specifications of interaction protocols. In this light and by assuming a similar abstraction for agents, we re-read the correctness properties for multi-agent systems, discussed in [36].

3 Design of Commitment-based Protocols

In this section we propose a representation of commitment-based protocols which encompasses a *constitutive* specification, defining the meaning of actions for all the agents in the system, and a *regulative* specification, constraining the possible evolutions of the social state (see Fig. 3). Instead, for what concerns players, we account both for the player's own actions and for its behavioral rules.

Definition 1 (Interaction protocol). *An interaction protocol* P *is a tuple* $\langle R, F, A, C \rangle$*, where* R *is a set of roles, identifying the interacting parties,* F *is a*

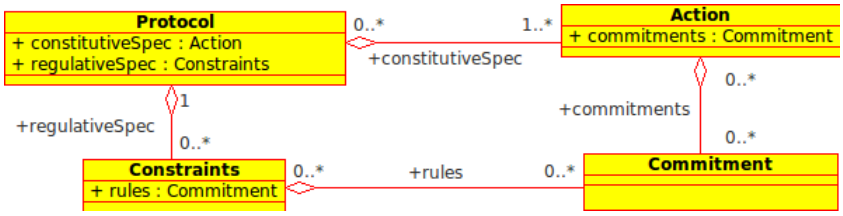

Fig. 1. Decoupling between constitutive (actions) and regulative (constraints) specifications.

set of fluents (including commitments) that can occur in the social state, A is a set of actions, and C is a set of constraints.

In words, the set of social actions A, defined on F and on R, forms the *constitutive specification* of the protocol, while the set of constraints C, defined on F and on R too, forms the *regulative specification* of the protocol.

Each role is identified by a unique label. Since both the constitutive and the regulative specifications are given also in terms of the roles involved in the actions or in the social commitments, it is possible to keep for each role, the set of the actions it can perform as well as the set of commitments it may be involved in as the interaction is carried on.

F is a set of fluents, where each fluent can alternatively be a commitment or a literal (fact), i.e. a positive or negative proposition that does not concern commitments and which contributes to the social state (they are the conditions that are brought about). The set F represents the domain model and defines the vocabulary used by all agents (through roles) to communicate in the context of the protocol. Currently F is a flat set but this representation can easily be structured by integrating an ontology layer into the domain model.

Constitutive Specification. It defines the meaning of actions in the very same way as it is done in [15], i.e. in terms of how it affects the social state by adding or removing fluents or by performing operations on the commitments, see [33, 42]. For instance, the action *priceRequest* of the Net Bill protocol (which is used as an example below) is given in this way:

$$priceRequest(c, m, goods) \textbf{ means } \textsc{create}(C(c, m, purchase(goods)))$$

i.e. its effect is to add to the social state a commitment $C(c, m, purchase(goods))$ by which the customer (role c) commits to a merchant (role m) to buy some goods. As we will see, the protocol includes also the action *rejectQuote*:

$$rejectQuote(c, m, goods, price) \textbf{ means}$$
$$rejectedQuote(goods, price) \wedge \textsc{delete}(C(c, m, purchase(goods)))$$

by which the customer rejects the quote received from the merchant. In this case, it deletes its commitment to buy. Commitment deletion is one of the basic operations on commitments, see [34].

An agent willing to play a role in a protocol, must understand the meaning of the social actions that are associated to the role at issue. In order to play the role, the agent must accept the meaning given to the social actions, which will be the same for all agents.

Regulative Specification. For the *regulative specification* C of an interaction protocol we propose a declarative, *constraint-based representation*. Due to the declarative nature of the specification, *any* evolution that respects the relations involving the specified fluents (including commitments) is allowed. Notice that constraints *do not* specify *which* actions should bring conditions about. This allows the decoupling between the constitutive and the regulative specifications, see also [7], Fig. 1 and the discussion in the Conclusions. The regulative specification follows the grammar:

$$C \rightarrow (Disj\ op\ Disj)^{+}$$
$$Disj \rightarrow Conj \vee Disj \mid Conj$$
$$Conj \rightarrow fluent \wedge Conj \mid fluent$$

C, see Def. 1, is a set of constraints of the form $A\ op\ B$, where A and B are formulas of fluents in disjunctive normal form and op is one of the operators in Table 1; *fluent* can be either a commitment or a fact. Such constraints rule the evolution of the social state by imposing specific patterns on how states can progress. In order to specify constraints it is necessary to define a proper language. One possible language, that we originally introduced in [7], is 2CL (the acronym stands for "Constraints among Commitments Language"), whose operators are summarized in Table 1.

Relation	*Positive*	*LTL meaning*	*Negative*	*LTL meaning*
Correlation	$a \bullet\!\!- b$	$\Diamond a \supset \Diamond b$	$a \bullet\!\!\not- b$	$\Diamond a \supset \neg\Diamond b$
Co-existence	$a \bullet\!\!-\!\!\bullet b$	$a \bullet\!\!- b \wedge b \bullet\!\!- a$	$a \bullet\!\!\not-\!\!\bullet b$	$a \bullet\!\!\not- b \wedge b \bullet\!\!\not- a$
Response	$a \bullet\!\!-\!\!\triangleright b$	$\Box(a \supset \Diamond b)$	$a \bullet\!\!\not-\!\!\triangleright b$	$\Box(a \supset \neg\Diamond b)$
Before	$a \rightarrow\!\!\bullet b$	$\neg b \mathsf{U} a$	$a \not\rightarrow\!\!\bullet b$	$\neg a \mathsf{U} b$
Cause	$a \bullet\!\!-\!\!\triangleright\!\!\bullet b$	$a \bullet\!\!-\!\!\triangleright b \wedge a \rightarrow\!\!\bullet b$	$a \bullet\!\!\not-\!\!\triangleright\!\!\bullet b$	$a \bullet\!\!\not-\!\!\triangleright b \wedge a \not\rightarrow\!\!\bullet b$
Premise	$a \bowtie\!\!- b$	$\Box(\bigcirc b \supset a)$	$a \bowtie\!\!\not- b$	$\Box(\bigcirc b \supset \neg a)$
Immediate response	$a \rightarrow\!\!\triangleright b$	$\Box(a \supset \bigcirc b)$	$a \not\rightarrow\!\!\triangleright b$	$\Box(a \supset \bigcirc \neg b)$

Table 1. 2CL operators and their semantics in LTL.

The names of the operators and the graphical format, used in Section 3.2, are inspired by ConDec [29]. In order to allow the application of reasoning techniques, e.g. to check if the on-going interaction is respecting the protocol, to build sequences of actions that respect the protocol, or to verify properties of the system, it is necessary to give the operators a semantics that can be reasoned about. To this aim, in this work we use *linear temporal logic* (LTL, [20]),

which includes temporal operators such as next-time ($\bigcirc$), eventually ($\Diamond$), always ($\Box$), weak until (U). Let us describe the various operators. For simplicity the descriptions are given on single fluents rather than on formulas.

Correlation: this operator captures the fact that in an execution where a occurs, also b occurs but there is no temporal relation between the two. Its negation means that if a occurs in some execution, b must not occur.

Co-existence: the mutual correlation between a and b. Its negation captures the mutual exclusion of a and b. Notice that in LTL the semantics of negated co-existence is equivalent to the semantics of negated correlation.

Response: this is a temporal relation, stating that if a occurs b must hold at least once afterwards (or in the same state). It does not matter if b already held before a. The negation states that if a holds, b cannot hold in the same state or after.

Before: this a temporal relation, stating that b cannot hold until a becomes true. Afterwards, it is not necessary that b becomes true. The negation of $a \multimap\!\!\bullet\, b$ is equivalent to $b \multimap\!\!\bullet\, a$.

Cause: this operator states that if a occurs, after b must occur at least once and b cannot occur before a. The negation states that if a occurs, b cannot follow it and if b occurs, a is not allowed to occur before.

Premise: is a stronger temporal relation concerning *subsequent* states, stating that a must hold in all the states immediately preceding one state in which b holds. The negation states that a must never hold in a state that immediately precedes one where b holds.

Immediate Response: it concerns *subsequent* states, stating that b must occur in all the states immediately following a state where a occurs. The negation states that b does not have to hold in the states immediately following a state where b holds.

Notice that the negated operators semantics (column 5) not always corresponds to the negation of the semantics of the positive operator (column 3). This is due to the intention of capturing the intuitive meaning of negations. We show this need by means of a couple of examples. For what concerns correlation, the negation of the formula in column 3 is $\Diamond a \wedge \neg\Diamond b$ is too strong because it says that a must hold sooner or later while b cannot hold. What we mean by negated coexistence, instead, that *if a becomes true* then b must not occur in the execution. For completeness, the semantics of negated correlation is not equivalent to the semantics of $a \bullet\!\!- \neg b$. For what concerns immediate response, by negating the semantics in column 3 we obtain $\Diamond(\bigcirc b \wedge \neg a)$ which says that b occurs in some state and a does not occur in the previous state. Instead, the intended meaning of the negation is that a does not have to hold in the states that precede those in which b holds (but b not necessarily have to hold). Analogous considerations can be drawn for the other operators. The choice of sticking to the intuitive semantics of the operators is done to give the user only seven basic operators. Had we defined the negated operators semantics by negating the semantics of the positive operators, we would have given the user forteen different operators.

3.1 Violation of constraints and of commitments

So, an interaction protocol includes a set of constraints, whose aim is to guarantee that all the interacting agents will achieve the expected results. This happens because by agreeing on the constraints they agree on the behavior they all will carry on. In this setting, does the violation of a constraint have the same nature of the violation of a commitment? According to Singh [33], commitments have a *normative* nature: an agent can freely decide if and when committing to do something but when it does it is obliged to fulfill the commitment. In particular, suppose a merchant has a nested commitment like this to rule a sequencing in commitments:

$$C(m, c, C(c, m, purchase(goods)), C(m, c, sold(goods, price)))$$

Here the merchant commits to take the commitment $C(m, c, sold(goods, price))$ if the customer commits to $C(c, m, purchase(goods))$. The problem is that, since the merchant is free to decide whether or not taking the outer commitment, the customer has *no guarantee* that its decision to buy the goods will be followed by the merchant's commitment to sell because there is no guarantee that the external commitment will be taken by the merchant. If, instead, we use one of our constraints, like this one:

$$C(c, m, purchase(goods)) \bullet\!\!\rightarrow\!\bullet\, C(m, c, sold(goods, price))$$

by which the commitment of the customer c to buy some goods $C(c,\ m,\ purchase(goods))$ imposes that the merchant m will sell the goods at some price $C(m, c,\ sold(goods, price))$, the customer has the guarantee that the merchant will take the commitment to sell the goods if it decides to buy. The customer knows this before starting the interaction due to the fact that the protocol is public and can use this information in order to decide whether to use the protocol.

Guarantees, however, are given only if constraints have a normative nature. So, the violation of a constraint, as well as the violation of a commitment, pushes the agent out of the protocol. Of course, in case of constraints being out of the protocol does not necessarily mean that the interaction with the others will be unsuccessful but only that the participants lose the guarantee that all the parties involved will achieve an effective result. Instead, by sticking to the constraints the agents waive part of their autonomy and this is done because considered advantageous w.r.t. interacting without rules.

3.2 An example: the Net Bill Protocol

The Net Bill Protocol [19] has the aim of satisfying the regulative necessities of the purchase of *electronic information goods* (simply goods, in the following) over a network. In this section we represent the part of the Net Bill Protocol that rules the interactions of a customer (or consumer) c, wishing to buy some information, and a merchant m. Intuitively, (1) the customer requests the price

of certain goods to the merchant, (2) the merchant answers by quoting the goods, (3) the customer can either accept or reject the quote, (4) if the customer accepts, it is sent the requested information goods in an encrypted form, (5) the customer pays the merchant, (6) the merchant sends the key for the decryption and the receipt of the payment to the customer. The constitutive specification of the protocol defines the meaning of actions in terms of the changes they make on the social state:

(a) $priceRequest(c, m, goods)$ **means** CREATE$(C(c, m, purchase(goods)))$
(b) $priceQuote(m, c, goods, price)$ **means** CREATE$(C(m, c, sold(goods, price)))\wedge$
CREATE$(C(m, c, sentEnc(goods)))$
(c) $acceptQuote(c, m, goods, price)$ **means** CREATE$(C(c, m, paid(goods, price)))$
(d) $rejectQuote(c, m, goods, price)$ **means** $rejectedQuote(goods, price)\wedge$
DELETE$(C(c, m, purchase(goods)))$
(e) $order(c, m, goods)$ **means** $purchase(goods)$
(f) $goodsDelivery(m, c, goods, price, key, receipt)$ **means**
$sold(goods, price) \wedge sentEnc(goods)\wedge$
CREATE$(C(m, c, sent(key))) \wedge$ CREATE$(C(m, c, sent(receipt)))$
(g) $pay(c, m, goods, price)$ **means** $paid(goods, price)$
(h) $sendKey(m, c, key)$ **means** $sent(key)$
(i) $sendReceipt(m, c, receipt)$ **means** $sent(receipt)$

The action *priceRequest* states the resolution (expressed by the commitment $C(c, m, purchase(goods))$) to buy certain goods from a merchant. This does not necessarily mean that the purchase will occur because the offer of the merchant can be rejected by the customer. By *priceQuote* the merchant commits to sell the requested goods at a certain price and to send them in an encrypted form. The acceptance of a quotation produces the commitment $C(c, m, paid(goods, price))$. Instead, rejecting the quote causes the deletion of the commitment to buy; the fluent *rejectedQuote* is also asserted. *order* asserts the fact $purchase(goods)$ and thus causes the discharge of the commitment to buy. *goodsDelivery* asserts that the goods have been sold at a certain price, it causes the discharge of the corresponding commitment to sell, it records that the encrypted goods have been sent, and records the commitment of the merchant to send the key as well as the receipt. The meaning of the other actions is simple and we do not describe it.

The regulative specification of the Net Bill protocol is given by these constraints (also shown in graphical format in Fig. 2):

c1: $C(c, m, purchase(goods)) \bullet\!\!\rightarrow\!\!\bullet\ C(m, c, sold(goods, price))\wedge$
$C(m, c, sentEnc(goods))$
c2: $C(m, c, sold(goods, price)) \wedge C(m, c, sentEnc(goods)) \multimap$
$rejectedQuote(goods, price)$ XOR
$(C(c, m, paid(goods, price)) \wedge purchase(goods))$
c3: $C(c, m, paid(goods, price)) \wedge purchase(goods) \bullet\!\!\rightarrow\!\!\bullet\ C(m, c, sent(key))\wedge$
$C(m, c, sent(receipt)) \wedge sold(goods, price) \wedge sentEnc(goods)$
c4: $C(m, c, sent(key)) \wedge C(m, c, sent(receipt)) \wedge sold(goods, price)\wedge$

$sentEnc(goods) \bullet\!\!\rightarrow\!\!\bullet\ paid(goods, price)$

c5: $paid(goods, price) \bullet\!\!\rightarrow\!\!\bullet\ sent(key)$

c6: $sent(key) \bullet\!\!\rightarrow\!\!\bullet\ sent(receipt)$

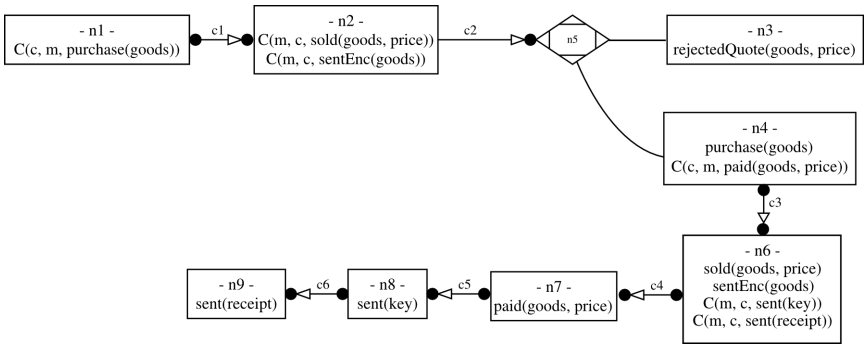

Fig. 2. Regulative specification of the Net Bill: boxes represent conjunctions of fluents, circles represent conjunctions of boxes, diamonds represent the XOR of boxes.

When a customer commits to buy some goods ($C(c, m, purchase(goods))$), the merchant will commit to sell the goods at a certain price and to send the encrypted information. This is specified as a *cause* ($\bullet\!\!\rightarrow\!\!\bullet$, constraint $c1$) relation. In Fig. 2 a rectangle containing many fluents (e.g. $n2$) represents a conjunction. These two fluents must hold *before* (constraint $c2$) the XOR relation between the fluent $rejectedquote(goods, price)$, which asserts that the quotation has been rejected, and the conjunction between the acceptance of the offer (fluent $purchase(goods)$) and the commitment of the customer to pay the agreed price (node $n4$). Notice that the customer is not obliged to reject or accept the quotation (and commit to pay), but once the merchant observes these two fluents in the social state it takes them as a guarantee that the customer will actually buy the information: so, the order of the execution of actions *order* and *goodsDelivery* is ruled indirectly. So, *afterwards* (constraint $c3$) it will send the encrypted information (node $n6$), confirm the price ($sold(goods, price)$), commit to send the key to decrypt the information and after (and only after) the customer has to pay (node $n7$). This condition *causes* (constraint $c5$) the dispatch of the key (node $n8$) and of the receipt (node $n9$).

Remark 1. Commenting the Net Bill, Chopra and Singh [14] criticize the use of finite state machines (FSM) because they lack of flexibility: the strict encoding of a request followed by an offer does not allow the merchant to take the initiative by advertising an attractive deal. To solve this limit they adopt commitment protocols, whose only constraint is that all commitments are discharged. In other

terms, they remove the specification of any sequence. Even though we agree that FSM are too rigid, in our opinion the aim of the Net Bill is to guarantee that the client will have an quotation when it requests it. By removing all sequencing relations flexibility is obtained at the cost of losing such guarantee. By substituting the *cause relation* ($\bullet\!\!\rightarrowtail\!\!\bullet$) in $c1$ with a *response relation* ($\bullet\!\!\rightarrowtail$), we obtain the desired flexible representation by the constraint $C(c, m, purchase(goods)) \bullet\!\!\rightarrowtail C(m, c, sold(goods, price)) \wedge C(m, c, sentEnc(goods))$, both allowing to start from an offer and keeping the guarantee that the customer receives the expected quotations when performing a request.

4 Correctness Properties of MAS

In Section 2 we assumed that *agents* are made of two components: a set of actions and a set of behavioral rules. Actions, see Fig. 3, are the basic building blocks of the agent's behavior. We do not make any assumption on how agents' actions are represented. They can have (or have no) preconditions to their execution, effects, or conditional effects on the agent's mental state. Their implementation is local to the agent. In this work, following [15, 36], we abstract away from the implementation details and represent the agents' actions as they are represented inside the interaction protocols. The same is done for the behavioral rules. In particular, we use the language 2CL, obtaining a representation that is homogeneous with the representation of the regulative specification of interaction protocols. This language is sufficiently expressive to abstractly represent any kind of behavior, as done for business processes by proposals like [29, 27, 25].

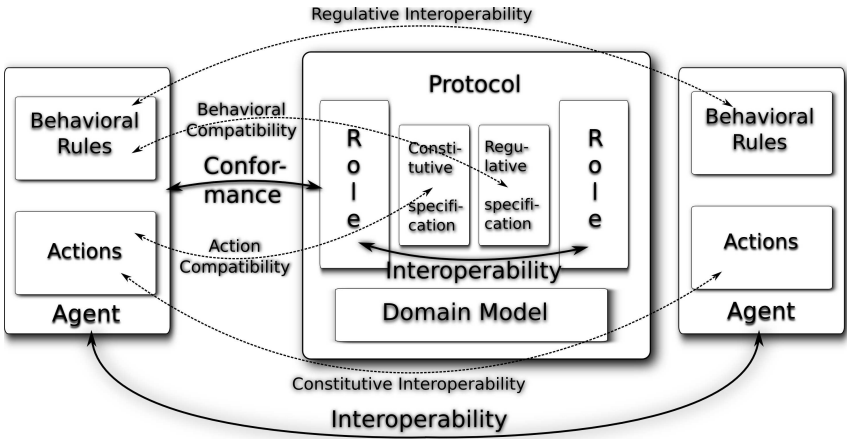

Fig. 3. Interoperability and conformance properties.

Considering protocols as *models of the desired interaction* allows one to devise the verification of many properties of the interaction, before any interaction takes

place. In order to compare this proposal to [36], in the following we discuss the properties of *interoperability* among agents/roles, and the *conformance* of agents to roles. We end the section with some considerations on the *refinement* property.

4.1 Agent Interoperability and Protocol Interoperability

Intuitively, a set of agents/roles is interoperable when it is stuck-free, i.e., when whatever point of interaction may be reached the system will not be blocked [6]; in other words, when the agents jointly meet the expectations they place on each other [36]. Singh and Chopra [36] consider interoperability as a conjunction of *liveness*, *safety* and *alignment*. *Liveness* means that the system will progress, i.e. it never happens that an agent waits for a message never sent by another agent. More generally, we say that liveness means that it never happens that an agent *waits for an action* that another agent is expected to execute, and that has never been performed. *Safety* means that an agent must be ready to handle messages that it receives. In other words, it is necessary to ensure that messages are sent in the order receivers wait for them. More generally, we say that agents must be *ready to re-act by performing an action* whenever this is expected by some other agent in the system.

Singh and Chopra propose to model liveness and safety by using *potential causality* of sends and receives of messages. The two properties are characterized by the compatibility among causal orders of sends and receives [24]. However, one of the key points of commitment protocols is that they allow ruling not only sends and receives but any social action whose meaning is agreed upon. For instance, the agents may agree upon the action *receiving goods* and *paying goods*, and the order expected by one of the two, say the customer, could be that goods will be paid only after reception. How to extend the notion of causality to the more general case? It would be necessary to express in some way the causal relations expected by the role players because they are not so obvious as with messages. Moreover, causality may be just one possible relation concerning the ordering of actions (other relations could be useful as shown in the Net Bill example). Even more importantly, agents rather than observing each other's *actions*, observe the *social state*, so it is more advisable that such relations concern the *evolution of the social state*. The regulative specification, being aimed at expressing constraints on the evolution of the social state, should indeed express such properties. We represent it by means of the language 2CL described in Section 3. In the case it is necessary to verify the interoperability of a set of protocol roles (also called *operability* in [36]), we suppose the regulative specification given as part of the protocol. Instead, in the case one wants to verify the interoperability of a set of agents, it is necessary that agents disclose, at least in a partial way, their own behavioral rules (same assumption of [36]). This, of course, supposing that they are already aligned on the meaning of their actions.

Alignment means that whenever an agent concludes to be the creditor of a commitment the corresponding debtor concludes that it is the debtor of the same commitment. The verification of alignment [15, 18] includes the verification of *constitutive interoperability* [17, 36]. Agents are constitutively interoperable

when they would agree about whatever commitments as might result from any messages they might exchange. Constitutive interoperability can be verified by reasoning on the *Actions* component of the involved agents, Fig. 3. Constitutive interoperability is included also in our proposal. In addition, since we foresee a decoupled representation of the regulative and of the constitutive specifications, it is possible to check also interoperability at the level of regulative specifications and to see if agents are, for instance, compatible at the level of actions but not at the level of behavior or the other way around. As a final observation, some alignment rules [15, 18] could actually be constraints specified in the regulative rules.

4.2 Conformance and Substitutability

The limit of verifying properties at the level of groups of individual agents is that the verifications can be done only when all such agents have been identified. The verifications are to be repeated whenever the group changes, i.e. when one of the agents leaves the group or is substituted by a new one. Protocols allow overcoming this limit. Given a protocol whose roles show the desired properties (mainly interoperability), it is possible to check agents one by one against the corresponding roles to see if they can interpret the role preserving the protocol properties (*substitutability*). When the protocol property of interest is interoperability, the substitutability is guaranteed by the *conformance* relation.

In our setting, we can specify two levels of conformance: conformance at the level of actions (*constitutive conformance*), and conformance at the level of behavior (*regulative conformance*). Constitutive conformance aims at verifying that an agent can establish a *count-as* relation between its own actions and the social actions. The reason is that when agents have to interact with one another in the context of a protocol they must be capable of providing an implementation for each of the actions accounted for by the role they want to play. Notice that it is not required that agents have actions that *exactly* match with the social actions [28, 8, 15]. It is not even required to have a 1-1 relation, so an agent may implement a social action by means of a sequence of its own actions.

By regulative conformance we mean the fact that the behavioral rules of the agent are not in conflict with the regulative specification of the protocol. Since, in general, the agent's behavioral rules can restrict the behaviors allowed by the regulative specification, it is also necessary to check that these restrictions do not impose constraints on the other players. In other words, the player is allowed to restrict its own behavior but it should not limit the freedom of the other agents, when they behave as specified by the protocol. For instance, in the Net Bill protocol the customer can continue to ask for offers until it receives one that it likes. If an agent playing the role of merchant has a constraint saying that it can produce only one offer, then, that agent limits the freedom of any agent playing the role of customer, which has the right (according to the protocol) to ask for as many offers as it likes. In general, the behavioral rules of an agent do not have to offend the autonomy of choice the protocol gives to the other agents.

One last property that it is interesting to mention is *compliance*, which amounts to verifying that an *execution* of an agent respects the expectations of the others, in the context of a protocol, e.g. [12]. As a difference with [36], the presence of the regulative specification of the protocol allows verifying along the run if the agent violates the constraints given by the protocol, without waiting to arrive to the end to see if all commitments are discharged. To put it simply, violations can be intercepted earlier.

4.3 Protocol Generalizations/Refinements

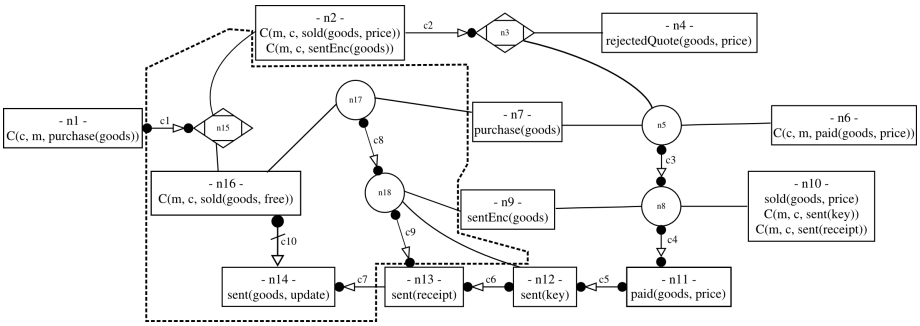

Fig. 4. Regulative specification of the generalized Net Bill: the dashed line highlights the constraints ruling the purchase of free goods. Boxes represent conjunctions of fluents, circles represent conjunctions of boxes, diamonds represent the XOR of boxes.

In interaction protocol engineering, it is often desirable to specialize or to generalize protocols so as to deal with more specific or wider contexts, e.g. [5]. The modular nature of our proposal allows the introduction of two levels of generalization/refinement: at the constitutive level, i.e. at the level of actions as in [36] as well as at the regulative level. In this latter case, we exploit the declarative nature of 2CL by producing broader or stricter sets of constraints. The so obtained protocols can be organized in a taxonomy. As an example, one might wish the Net Bill to handle in a special way the case in which some goods are free: when this happens no payment is requested. The implementation can be obtained (Figure 4) as a generalization of the Net Bill (Figure 2) by enriching the constraints with a XOR that introduces a new branch for the free-goods case:

c1: $C(c, m, purchase(goods)) \bullet\!\!\rightarrow\!\!\bullet\ C(m, c, sold(goods, free))$ XOR
$(C(m, c, sold(goods, price)) \wedge C(m, c, sentEnc(goods)))$

c7: $sent(receipt) \rightarrow\!\!\bullet\ sent(goods, update)$

c8: $C(m, c, sold(goods, free)) \wedge purchase(goods) \bullet\!\!\rightarrow\!\!\bullet$
$sentEnc(goods) \wedge sent(key)$

c9: $sentEnc(goods) \wedge sent(key) \bullet\!\!\rightarrow\!\!\bullet\ sent(receipt)$

c10: $C(m, c, sold(goods, free)) \bullet\!\!\not\triangleright sent(goods, update)$

Constraint $c1$ was modified to explicitly tackle the free-goods case. Constraint $c8$ states that goods are sent together with the key if both the merchant has offered it for free and the customer has accepted to purchase it (fluent *purchase(goods)*). The receipt is sent at the end ($c9$). Finally, ($c10$) if some information updates become available, the merchant does not send it if the customer bought the information for free.

5 Conclusion and Related Works

This work proposes a commitment-based approach to protocol definition, that is inspired by the work of Singh and colleagues [14, 34, 42, 41, 15, 36], which introduces an explicit representation of both constitutive and regulative specifications in the spirit of [32, 11]. Both specifications are given in a declarative way. The constitutive specification gives the meaning of the social actions, in terms of operations on the social state, as in [15]. The regulative specification is given as a set of constraints on the evolution of the social state expressed in 2CL. The semantics of 2CL is grounded on LTL. The proposed approach keeps the flexibility of commitment-based protocols, *indirectly* ruling the execution of the actions. The regulative specification is introduced because, in our opinion and we have tried to prove it in this analysis, the mere constitutive specification of actions is not sufficient, because agents have a behavior and this behavior makes them use actions in specific orders. This order gives actions a supplementary meaning, that must be taken into account in the interaction with the others. Since protocols are supposed to give the shared meaning of actions it is necessary that they account also for meaning given by specific ways of using actions, i.e. by patterns of behavior. By our proposal and by exploiting a declarative language, we have proved that it is possible to express this meaning without losing the flexibility of commitment-based protocols. We do this by putting constraints on the evolution of the social state and not on actions because, as shown in [7], this allows a greater modularity in the specification, with the advantages discussed in Section 2. In this paper, we have also shown that the introduction of a regulative specification does not compromise the proof of interoperability properties but rather it allows finer verifications.

Chopra and Singh [17] recognize the distinction between constitutive and regulative specifications in the definition of commitment-based protocols but focus their work on the constitutive component only, see [15, 16, 36]. When there is the need to constrain the behavior of agents, they use preconditions to the (non-)executability of the actions. This solution (which is adopted also by other works, like [23, 41, 42, 15, 39]) is characterized by a *strong localization* of the regulative specification; the constitutive and the regulative specifications are indistinguishable (being both based on actions) and action become dependant on the protocol they are used in. This limits the openness of the system and in particular complicates the re-use of software (the agents' actions). A too tight relation to actions can be ascribed also to [35], although in this work

it is possible to recognize the introduction of a regulative specification, based on the *before* relation. Such relations are, however, applied to events/actions. Fornara and Colombetti [21, 22] recognize the need of a regulation of the flow of execution but adopted *interaction diagrams* in the definition of agent interaction protocols. Interaction diagrams force the ordering of action executions, loosing, in our opinion the flexibility aimed at by the adoption of commitments.

Outside the Agents research area, Pesic and van der Aalst [29] propose an approach that is radically opposite to the one by Chopra and Singh: it totally lacks of a constitutive component but it uses the declarative language ConDec for representing business processes (which, though not exactly interaction protocols, specify the expected behavior of a set of interacting parties by constraining the execution of their tasks). The regulative rules are a first-class element of the protocol which are given by means of an ad hoc *declarative* language. They are not local to single actions, rather they are constraints that rule the flow of activity execution (activity in UML sense). In [27, 13, 25], the authors use this approach to specify interaction protocols and service choreographies. To this aim, they integrate ConDec with SCIFF thus giving a semantics to actions that is based on *expectations*. Still, however, these proposals show a too tight connection between the regulative rules and actions because such rules define temporal constraints over actions (events). With respect to [37], our proposal does not handle time explicitly so we cannot yet represent and handle timeouts and also compensation mechanisms. We plan to tackle these issues in future work.

Acknowledgements

The authors would like to thank the reviewers for the helpful comments. This research has partially been funded by "Regione Piemonte" through the project ICT4LAW.

References

1. *Proc. of the 2nd Multi-Agent Logics, Languages, and Organisations Federated Workshops*, volume 494 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.
2. *Declarative Agent Languages and Technologies VII, 7th Int. Workshop, DALT 2009*, volume 5948 of *Lecture Notes in Computer Science*. Springer, 2010.
3. M. Alberti, D. Daolio, P. Torroni, M. Gavanelli, E. Lamma, and P. Mello. Specification and verification of agent interaction protocols in a logic-based system. In *Proc. of the SAC 2004*, pages 72–78. ACM, 2004.
4. R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *CONCUR*, volume 1466 of *LNCS*, pages 163–178. Springer, 1998.
5. L. Astefanoae and F. S. de Boer. Model-checking agent refinement. In *Proc. of AAMAS '08*, pages 705–712, 2008.
6. M. Baldoni, C. Baroglio, A.K. Chopra, N. Desai, V. Patti, and M. P. Singh. Choice, Interoperability, and Conformance in Interaction Protocols and Service Choreographies. In *Proc. of AAMAS'09*, pages 843–850, 2009.

7. M. Baldoni, C. Baroglio, and E. Marengo. Constraints among commitments: Regulative specification of interaction protocols. In *Proc. of Agent Communication*, 2010. In conjunction with AAMAS 2010.
8. M. Baldoni, C. Baroglio, V. Patti, and C. Schifanella. Conservative re-use ensuring matches for service selection. In *Proc. of EUMAS 2008*, Bath, UK, 2008.
9. G. Boella and L. W. N. van der Torre. Regulative and constitutive norms in normative multiagent systems. In *Proc. of KR*, pages 255–266. AAAI Press, 2004.
10. M. Bravetti and G. Zavattaro. A theory of contracts for strong service compliance. *Mathematical Structures in Computer Science*, 19(3):601–638, 2009.
11. C. Cherry. Regulative rules and constitutive rules. *The Philosophical Quarterly*, 23(93):301–315, 1973.
12. F. Chesani, P. Mello, M. Montali, F. Riguzzi, M. Sebastianis, and S. Storari. Checking Compliance of Execution Traces to Business Rules. In *Business Process Management Workshops*, volume 17 of *LNBIP*, pages 134–145. Springer, 2008.
13. F. Chesani, P. Mello, M. Montali, and P. Torroni. Verifying a-priori the composition of declarative specified services. In *MALLOW* [1].
14. A. K. Chopra and M. P. Singh. Nonmonotonic Commitment Machines. In *Workshop on Agent Communication Languages*, volume 2922 of *Lecture Notes in Computer Science*, pages 183–200. Springer, 2003.
15. A.K. Chopra. *Commitment Alignment: Semantics, Patterns, and Decision Procedures for Distributed Computing*. PhD thesis, North Carolina State University, Raleigh, NC, 2009.
16. A.K. Chopra and M. P. Singh. Contextualizing commitment protocol. In *Proc. of AAMAS'06*, pages 1345–1352. ACM, 2006.
17. A.K. Chopra and M. P. Singh. Constitutive interoperability. In *Proc. of AAMAS'08*, pages 797–804, 2008.
18. A.K. Chopra and M. P. Singh. Multiagent commitment alignment. In *Proc. of AAMAS'09*, pages 937–944, 2009.
19. B. Cox, J. D. Tygar, and M. Sirbu. NetBill Security and Transaction Protocol. In *WOEC'95: Workshop on Electronic Commerce*, 1995.
20. E. A. Emerson. Temporal and Modal Logic. In *Handbook of Theoretical Computer Science*, volume B, pages 997–1072. Elsevier, 1990.
21. N. Fornara. *Interaction and Communication among Autonomous Agents in Multiagent Systems*. PhD thesis, Università della Svizzera italiana, Facoltà di Scienze della Comunicazione, June 2003.
22. N. Fornara and M. Colombetti. A Commitment-Based Approach To Agent Communication. *Applied Artificial Intelligence*, 18(9-10):853–866, 2004.
23. L. Giordano, A. Martelli, and C. Schwind. Specifying and verifying interaction protocols in a temporal action logic. *J. Applied Logic*, 5(2):214–234, 2007.
24. L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communication of the ACM*, pages 558–565, 1978.
25. Montali M., M. Pesic, W.M. P. van der Aalst, F. Chesani, P. Mello, and S. Storari. Declarative specification and verification of service choreographies. *ACM Transactions on the Web*, 2009.
26. A. U. Mallya and M. P. Singh. Introducing preferences into commitment protocols. In *AC*, volume 3859 of *LNCS*, pages 136–149. Springer, 2006.
27. M. Montali. *Specification and Verification of Declarative Open Interaction Models - A Logic-based framework*. PhD thesis, University of Bologna, 2009.
28. A. Moormann Zaremski and J.M. Wing. Specification matching of software components. *ACM Transactions on SEM*, 6(4):333–369, 1997.

29. M. Pesic and W. M. P. van der Aalst. A Declarative Approach for Flexible Business Processes Management. In *Proc. of. Business Process Management Workshops*, volume 4103 of *LNCS*, pages 169–180. Springer, 2006.
30. S. K. Rajamani and J. Rehof. Conformance checking for models of asynchronous message passing software. In *CAV*, volume 2404 of *LNCS*, pages 166–179. Springer, 2002.
31. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Series in Artificial Intelligence. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.
32. J. Searle. *Speech Acts*. Cambridge University Press, 1969.
33. M. P. Singh. An ontology for commitments in multiagent systems. *Artif. Intell. Law*, 7(1):97–113, 1999.
34. M. P. Singh. A social semantics for agent communication languages. In F. Dignum and M. Greaves, editors, *Issues in Agent Communication*, volume 1916 of *LNCS*, pages 31–45. Springer, 2000.
35. M. P. Singh. Distributed enactment of multiagent workflows: temporal logic for web service composition. In *AAMAS*, pages 907–914. ACM, 2003.
36. M. P. Singh and A. K. Chopra. Correctness properties for multiagent systems. In *DALT* [2], pages 192–207.
37. P. Torroni, F. Chesani, P. Mello, and M. Montali. Social commitments in time: Satisfied or compensated. In *DALT* [2], pages 228–243.
38. M. B. van Riemsdijk. *Cognitive Agent Programming, a Semantic Approach*. PhD thesis, Dutch Research School for Information and Knowledge Systems, Utrecht University, the Netherlands, 2006.
39. M. Winikoff, W. Liu, and J. Harland. Enhancing commitment machines. In *Proc. of DALT*, volume 3476 of *LNCS*, pages 198–220, 2004.
40. M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2002.
41. P. Yolum and M. P. Singh. Commitment machines. In *Proc. of ATAL*, volume 2333 of *LNCS*, pages 235–247. Springer, 2001.
42. P. Yolum and M. P. Singh. Designing and executing protocols using the event calculus. In *Agents*, pages 27–28, 2001.