Auton Agent Multi-Agent Syst (2007) 14:165–186
DOI 10.1007/s10458-006-0011-1

Conversational semantics sustained by commitments

Roberto A. Flores · Philippe Pasquier · Brahim Chaib-draa

Published online: 9 June 2006

Abstract We propose an operational model that combines message meaning and conversational structure in one comprehensive approach. Our long-term research goal is to lay down principles uniting message meaning and conversational structure while providing an operational foundation that could be implemented in open computer systems. In this paper we explore our advances in one aspect of meaning that in theories of language use is known as "signal meaning", and propose a layered model in which the meaning of messages can be defined according to their fitness to advance the state of joint activities. Messages in our model are defined in terms of social commitments, which have been shown to entice conversational structure.

Keywords Social commitments · Interaction protocols · Formal specification · Multiagent system modeling and design

1 Introduction

Traditional agent communication language approaches define the meaning and sequencing of messages independently of each other by using mental states and conversation protocols, respectively. Although successful for many years in advancing the state of the art in agent communication, these approaches have failed to support conversations in open systems, which mandate that the mental states of agents be verified

R. A. Flores (✉)
Department of Physics, Computer Science & Engineering,
Newport News, VA 23606, USA
e-mail: flores@pcs.cnu.edu

P. Pasquier · B. Chaib-draa
Department of Computer Science & Software Engineering,
Sainte-Foy, Quebec, Canada G1K 7P4

P. Pasquier
e-mail: pasquier@damas.ift.ulaval.ca

B. Chaib-draa
e-mail: chaib@damas.ift.ulaval.ca

to abide with the messages they utter without assuming their goodwill to do so [18], and that protocols bear a correlation between the meaning of messages and their use in conversations [15]. Following these principles would allow both the engineering of protocols at design time and the analysis of conversations at runtime, thus enabling interactions between rule-based and protocol-based agents.

Recent research trends propose that the meaning and connectedness of messages should be defined using social commitments [2, 7, 9, 12]. We follow these trends and propose a model in which the meaning of messages is expressed according to their use in conversations advancing the state of commitments and the actions these commitments entail in the joint activities in which agents participate.

1.1 Research contribution

Coherent interactions have been known to require shared knowledge on the meaning of messages and their connectedness in conversations [4].

In this paper, we present our initial steps towards merging these two aspects under the umbrella of a social commitment-based approach. In the past, we have demonstrated that agreement to advance the state of commitments entails conversational structure [7]. Here, we argue that messages are meaningful only if they advance the state of the commitments that exist in their context of utterance. This is the premise of "signal meaning": messages are meaningful if they entail effects that are independent of the intentions behind their utterance.

The present work fits within our long-term goal of supporting agent interactions in open environments. Achieving this goal requires identifying theoretical principles that can be used to define a practical framework upon which interactions can be built. To this end, all notions in our work are formalized using Object-Z [20], which is an object-oriented extension of the Z formal specification language [5]. The main advantage of using this language is that it is reasonably straightforward for translating definitions into object-oriented implementations. A brief overview of Object-Z is included as an annex to this paper.

In brief, the primary contribution of our research is to advance the state-of-the-art in agent communication languages by proposing an operational model that combines message meaning and conversational structure in one comprehensive and implementable approach.

1.2 Language use: speaker and signal meaning

Theories of language use [3] define two complementary types of meaning: *speaker's meaning*, which is based on the use of messages for the communication of intent, and *signal meaning*, which is based on the use of messages as coordinating devices advancing the state of activities.

We drew inspiration from this latter type of meaning and conceptualized messages as coordinating devices advancing conversations that establish social commitments that bring about actions advancing joint activities. Following this view, we explore the possibility that the meaning of messages could be based on the context that is considered at four cumulative layers of abstraction, which are illustrated in Fig. 1:

- *Compositional* layer, where meaning can be ascribed to messages outside any conversational context, taking into account only the relationships between the instances they contain. For example, a message where the prospective speaker is

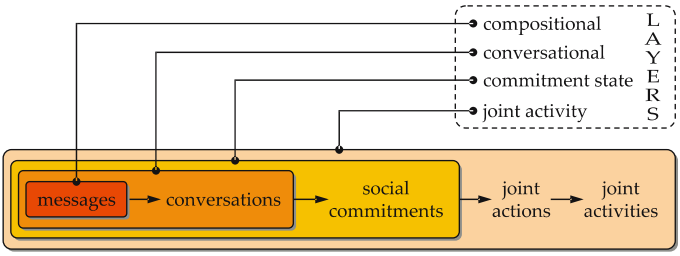

Fig. 1 Message semantics layers

both the actor of an action and the agent responsible for making the action happen could be interpreted as an offer that, when spoken, would commit the speaker to this course of action.

- *Conversational* layer, where meaning is based on the fitness of utterances to advance conversations. At this point, the utterance of messages should only be considered meaningful if they fit within the structure of new or ongoing conversational threads.
- *Commitment state* layer, where meaning is given to messages according to the state of the commitments they seek to advance. For example, an utterance attempting to retract a commitment should only be meaningful if such a commitment is mutually known by agents and if it is in a state from which it could be retracted.
- *Joint activity* layer, where meaning is given according to the contribution of messages to advance joint activities. That is, a meaningful message would be one that either continues an ongoing activity between agents, or attempts to begin an activity that is compatible with the roles that agents are portrayed to play in the system.

1.3 Overview

To support our model and present the rationale behind it, Section 2 briefly describes social commitments and their life cycle, including the states in which commitments exist and the transitions that could take them through these states. We emphasize transitions that are accomplished through the utterance of messages. Section 3 describes the fundamentals of our model, including the notion of agents as holders of images of other agents, where images store the messages exchanged between agents and the commitments they establish and discharge. In this context we define the notions of shared utterances and shared commitments. Section 4 builds upon these notions and presents the layers where message meaning is incrementally given according to the context these layers define. After presenting an example illustrating the applicability of the model to structure conversations (Section 5) and define message meaning (Section 6), we conclude with a few remarks on related and future work (Sections 7 and 8).

2 Social commitments

Social commitments [1, 17, 21] have been proposed as a way to raise expectations about agent performances. Commitments are defined as engagements in which an

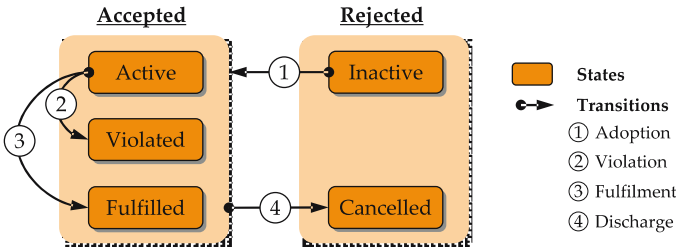

Fig. 2 Social commitment states and transitions

agent (called the debtor) is responsible relative to another agent (called the creditor) for satisfying a given condition.

Commitments have a life cycle made of states and transitions. As shown in Fig. 2, a commitment can be either *accepted* or *rejected* according to whether or not agents are engaged in it. If accepted, a commitment can be *active*, *violated* or *fulfilled*; if rejected, it can be either *inactive* or *cancelled*. Commitments can move between states through four transition types: *adoption*, where an inactive commitment becomes accepted; *violation* and *fulfillment*, where an active commitment becomes violated or fulfilled, respectively; and *discharge*, where an accepted commitment becomes cancelled. Initially, all commitments are inactive, but can become accepted upon adoption. Adopted commitments are classified as active, violated or fulfilled according to the state of achievement of their conditions of satisfaction (whether these conditions could be met, cannot be met, or have been met, respectively), and can become cancelled upon discharge. It is worth noticing that violation and fulfillment depend on the conditions of satisfaction, and that adoption and discharge are accomplished through (conversational) agreement. We model only conversational transitions (adoption and discharge) and assume that transitions based on the conditions of satisfaction (violation and fulfillment) are carried out automatically.

2.1 Social commitment operations

We define two operations targeted to commitment stores: adding and deleting.[1] To support agent autonomy, each agent is the only one that can directly add and delete commitments from its commitment store. However, since commitments are multi-agent constructs, involved agents can follow communicational conventions through which they seek to concurrently modify their commitment stores. This is a view held in argumentation [21], where utterances evidence commitments, either explicitly or implicitly. As described in the following sections, we analyze the case in which commitments are explicitly used in communication.

2.2 Social commitment messages

Messages are modelled as communicative actions in which a speaker sends an addressee a (non-empty) set of conversational tokens.

Agents can use the following four tokens to seek agreement to concurrently modify their commitment stores:

[1] That an agent adopts a commitment means that the commitment is added to the commitment store of the agent. Likewise, discharging a commitment indicates that the commitment is deleted from the agent's commitment store.

- *propose*, which indicates commitment operations upon which agreement is sought, and a time interval by which a reply is expected.
- *accept* and *reject*, which are replies indicating an acceptance or rejection to apply proposed operations to agents' commitment stores.
- *counter*, which rejects previously proposed commitment operations while simultaneously proposing others to be considered instead.

One last token, named *inform*, is used to communicate data.

2.3 Conversational transitions

We use a simple interaction protocol called *protocol for proposals* (*pfp*) [6] as a vehicle that agents can use to seek explicit agreement to adopt or discharge social commitments from their commitment stores.

As shown in Fig. 3, the protocol starts with a proposal (i.e., a message with a *propose* token in it) from agent a to agent b. This message can be followed (before the expiration of a reply deadline) by either pattern α or pattern β. Pattern α indicates that either agent b sends an accepting message to agent a, or that the interaction continues with pattern β (but with agents a and b's participatory roles inverted, i.e., the agent that in pattern α was agent a will be agent b in pattern β, and likewise for agent b). Pattern β indicates that agent a sends a rejection or a counterproposal message to agent b, in which case the interaction follows (before the expiration of a reply deadline) by either pattern α or pattern β. All replies except a counterproposal terminate the protocol; and when an acceptance is issued, both a and b apply the proposed and accepted operations to their commitment stores.

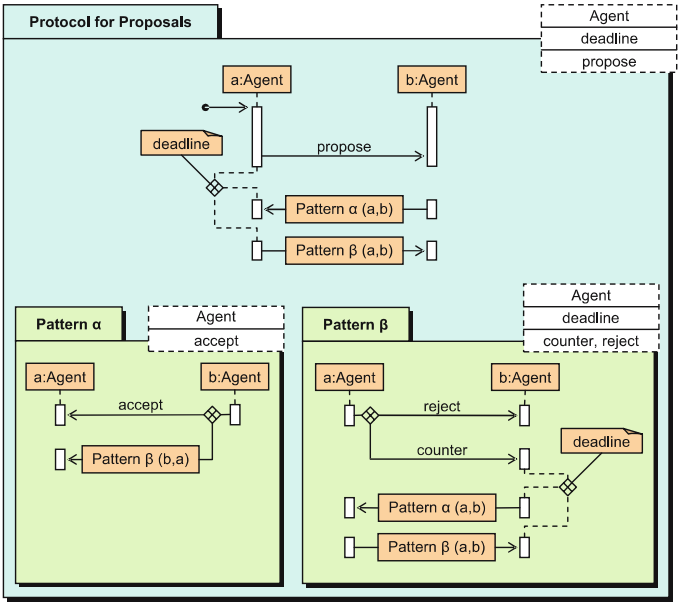

Fig. 3 The *protocol for proposals*

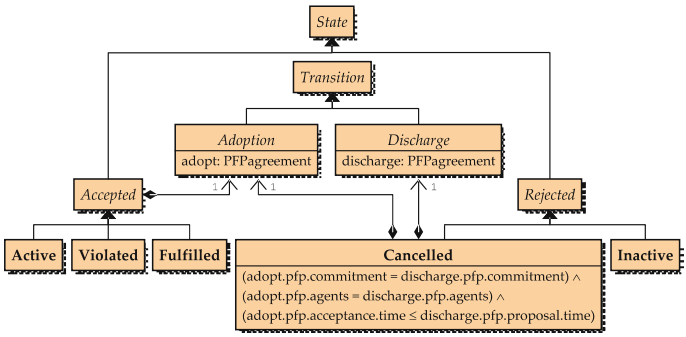

Fig. 4 Class diagram of social commitment states and *pfp* transitions

3 Fundamentals

3.1 States, transitions and agreement

As explained earlier, we are only concerned with two commitment state transitions: adopting and discharge, and then only when these transitions are agreed upon conventional communicational events.

In this analysis, we specify utterances as events marking the occurrence of messages at a certain moment in time, and agreement as a pair of sequential *pfp* utterances where the first utterance has a propose token and the second utterance has an accept token, both with identical commitment operations, and where these utterances were spoken and heard and then heard and spoken (in that order) by the same two agents. This agreement is defined as *pfp-agreement* (not shown).

Figure 4 shows the class hierarchy of commitment states and *pfp*-based transitions, where accepted commitments are adopted through agreement, and where cancelled commitments are first adopted and then discharged (also through agreement) by the same two agents.

3.2 Images and agents

We conceptualize agents as image holders, and images as agent representations storing utterances and commitments. To avoid complex constructs (such as agents holding images that hold images, *ad infinitum*, of other agents), we limit these definitions through the following two properties: agents only capture utterances in which they are either speaker or addressee (thus circumventing intricate ascriptions, such as agents being thought to have witnessed other agents' communications); and, communications are reliable (which means that the issuing of an utterance implies that speaker and addressee are aware that it happened). These properties help us capture the shared state of utterances in the same spirit as that of *shared-basis common ground* [3].

As shown below, images are defined as repositories of utterances and commitments, where each commitment is associated with a state, and where all stored utterances and commitments involve the same agent as a speaker or addressee, and as a creditor or debtor, respectively.

$$CommitmentState == SocialCommitment \times \downarrow State$$

SocialCommitmentHolder

$$
\begin{array}{l}
store : Time \to \mathbb{P}\, CommitmentState \\
holder : \downarrow Agent \\
\hline
\forall\, time : Time;\ sc : SocialCommitment \mid \\
\quad sc \in \mathrm{dom}(store(time)) \\
\bullet\ holder \in \{sc.creditor, sc.debtor\}
\end{array}
$$

UtteranceHolder

$$
\begin{array}{l}
witnessed : \mathbb{P}\, Utterance \\
holder : \downarrow Agent \\
\hline
\forall\, u : witnessed \bullet holder \in u.speechact.performers
\end{array}
$$

Image

$$
\begin{array}{l}
SocialCommitmentHolder \\
UtteranceHolder
\end{array}
$$

Agents are specified as self-aware (at least with respect to utterances and commitments) image holders that keep a consistent record of utterances and commitments. As shown below, this means that every agent keeps an image of itself, and that all utterances and commitments between an agent and any other agent are recorded in both the image that the holder keeps of itself and the image it keeps of the other agent.

ImageHolder

$$
\begin{array}{l}
awareof : \downarrow Agent \to Image \\
\hline
\forall\, image : Image;\ agent : \downarrow Agent \mid \\
\quad image = awareof(agent) \\
\bullet\ agent = image.holder
\end{array}
$$

Agent

ImageHolder

$$
\begin{array}{l}
\forall\, image_1, image_2 : Image;\ agent_2 : \downarrow Agent \mid \\
\quad image_1 = awareof(self) \wedge \\
\quad image_2 = awareof(agent_2) \\
\bullet\ (\forall\, u_1 : image_1.witnessed;\ u_2 : image_2.witnessed \mid \\
\qquad agent_2 \in u_1.speechact.performers \wedge \\
\qquad self \in u_2.speechact.performers \\
\quad \bullet\ u_1 \in image_2.witnessed \wedge \\
\qquad u_2 \in image_1.witnessed) \wedge \\
\quad (\forall\, time : Time \\
\quad \bullet\ \forall\, sc_1 : \mathrm{dom}(image_1.store(time));\ sc_2 : \mathrm{dom}(image_2.store(time)) \mid \\
\qquad agent_2 \in \{sc_1.debtor, sc_1.creditor\} \wedge \\
\qquad self \in \{sc_2.debtor, sc_2.creditor\} \\
\quad \bullet\ sc_1 \in \mathrm{dom}(image_2.store(time)) \wedge \\
\qquad sc_2 \in \mathrm{dom}(image_1.store(time)))
\end{array}
$$

Given the principle of reliable communications, it would be possible to infer that for all utterances, both speaker and addressee will have an image of themselves in which they share the utterance. Also, that agents follow agreement conventions based on utterances means that agents should also keep a consistent record of commitments. The shared state of utterances and social commitments is defined next.

3.3 Sharing utterances and commitments

An utterance is shared between its speaker and addressee if these two agents are aware that the utterance has been witnessed by both of them—which is a premise that holds true given the assumption that communications are reliable. Thus, an utterance is shared if its speaker and addressee hold images in which they have witnessed its occurrence.

$$
\begin{array}{l}
\mathit{SharedUtterances} : \downarrow\mathit{Agent} \times \downarrow\mathit{Agent} \rightarrow \mathbb{P}\,\mathit{Utterance} \\
\hline
\forall\, \mathit{agent}_1, \mathit{agent}_2 : \downarrow\mathit{Agent} \\
\bullet\ \mathit{SharedUtterances}(\mathit{agent}_1, \mathit{agent}_2) = \\
\quad \{u : \mathit{Utterance} \mid (u.\mathit{speechact.performers} = \{\mathit{agent}_1, \mathit{agent}_2\}) \wedge \\
\quad\ (\forall\, \mathit{agent} : u.\mathit{speechact.performers} \\
\quad\quad \bullet\ \exists\, \mathit{speaker}, \mathit{addressee} : \mathit{Image} \mid \\
\quad\quad\quad \mathit{speaker} = \mathit{agent.awareof}(u.\mathit{speechact.speaker}) \wedge \\
\quad\quad\quad \mathit{addressee} = \mathit{agent.awareof}(u.\mathit{speechact.addressee}) \\
\quad\quad\quad \bullet\ u \in \mathit{speaker.witnessed} \wedge \\
\quad\quad\quad\quad u \in \mathit{addressee.witnessed})\}
\end{array}
$$

Likewise, a commitment is shared between its creditor and debtor if these agents hold images recording the commitment in the same state. In addition, having a shared commitment implies that agents also share the utterances that brought the commitment to its shared state.

$$
\begin{array}{l}
\mathit{SharedCommitments} : \mathit{Time} \times \downarrow\mathit{Agent} \times \downarrow\mathit{Agent} \rightarrow \mathbb{P}\,\mathit{CommitmentState} \\
\hline
\forall\, \mathit{time} : \mathit{Time};\ \mathit{agent}_1, \mathit{agent}_2 : \downarrow\mathit{Agent} \\
\bullet\ \mathit{SharedCommitments}(\mathit{time}, \mathit{agent}_1, \mathit{agent}_2) = \\
\quad \{\mathit{sc} : \mathit{CommitmentState} \mid \\
\quad\ \forall\, \mathit{agent} : \{\mathit{agent}_1, \mathit{agent}_2\} \\
\quad\ \bullet\ \exists\, \mathit{image}_1, \mathit{image}_2 : \mathit{Image} \mid \mathit{image}_1 = \mathit{agent.awareof}(\mathit{agent}_1) \wedge \\
\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\ \ \mathit{image}_2 = \mathit{agent.awareof}(\mathit{agent}_2) \\
\quad\quad \bullet\ \mathit{sc} \in \mathit{image}_1.\mathit{store}(\mathit{time}) \wedge \mathit{sc} \in \mathit{image}_2.\mathit{store}(\mathit{time})\}
\end{array}
$$

4 Layers of interpretation

Based on these notions, we define an incremental layered model to ascribe meaning to messages. Thus far, we identify four layers: the *compositional* layer, where meaning is based on relationships between the constituent instances of messages; the *conversational* layer, where meaning is based on the occurrence of messages in conversations seeking agreement to advance the state of commitments; the *commitment state* layer, where meaning is based on the current state of targeted commitments; and the *joint*

activity layer, where meaning is based on the use of messages to advance actions in joint activities.

4.1 The compositional layer

Definitions at this layer support the categorization of messages based on the identity and relations between their instance components. These definitions are independent of the occurrence of messages as utterances, and thus allow classifying messages outside the scope of conversations.

In the context of the *pfp*, meaning is given to a message based on the type of conversational tokens, commitment operations and actions it contains, and on the relationship between the agents involved as speaker and addressee in its utterance, as creditor and debtor in its commitment, and as actors in its commitment actions.

For example, a message could be labelled as a *proposal* (as defined by *ToPropose* below) if it contains a propose token, and if the speaker and addressee are the creditor and debtor of the proposed commitment.

$$
\begin{array}{|l}
ToPropose : ToSpeak \rightarrow \mathbb{P} \downarrow Propose \\
\hline
\forall s : ToSpeak \\
\bullet\ ToPropose(s) = \\
\quad \{p : \downarrow Propose \mid (p \in s.tokens) \wedge \\
\qquad\qquad (s.performers = \{p.proposing.commitment.creditor, \\
\qquad\qquad\qquad p.proposing.commitment.debtor\})\}
\end{array}
$$

Similarly, a message is a *reply* (as defined by *ToReply*) if it contains a reply token (i.e., either an accept, reject or counter token), if the speaker and addressee are the creditor and debtor of the commitment, and if there is no other reply token in the message that refers to the replied commitment operation (thus avoiding cases in which the same message could have several tokens replying to the same *pfp* conversation).

$$
\begin{array}{|l}
ToReply : ToSpeak \rightarrow \mathbb{P} \downarrow Reply \\
\hline
\forall s : ToSpeak \\
\bullet\ ToReply(s) = \\
\quad \{r : \downarrow Reply \mid \\
\quad\ (r \in s.tokens) \wedge \\
\quad\ (s.performers = \{r.replying.commitment.creditor, \\
\qquad\qquad r.replying.commitment.debtor\}) \wedge \\
\quad\ (\nexists r_1 : \downarrow Reply \mid \\
\qquad r_1 \in s.tokens \wedge r_1 \neq r \\
\qquad \bullet\ r_1.replying = r.replying)\}
\end{array}
$$

These definitions can be used for specifying other messages with more refined meanings. For example, *acceptance* and *rejection* could be specified as *replies* with an accept and a reject token, respectively; *offer* and *request* can be defined as proposals where the speaker and the addressee (respectively) are the debtor of the commitment. As shown below, a *request* is defined as a proposal where the speaker is the creditor and the addressee is the debtor of the proposed commitment.

$$ToRequest : ToSpeak \to \mathbb{P} \downarrow Propose$$

$$\begin{aligned}
&\forall s : ToSpeak \\
&\bullet\ ToRequest(s) = \\
&\quad \{p : ToPropose(s) \mid \\
&\qquad s.speaker = p.proposing.commitment.creditor \land \\
&\qquad s.addressee = p.proposing.commitment.debtor\}
\end{aligned}$$

4.2 The conversational layer

This layer builds upon the *compositional* layer, and indicates the meaning of messages in the context of ongoing or potential conversations. Definitions take into account the time when utterances are issued, whether these utterances are shared, and whether they begin or could fit within one of the already started *pfp* conversations.

There are certain conditions that utterances should fulfil to be part of *pfp* conversations. While some conditions are trivial (e.g., proposals should indicate a time frame in which a reply is expected), some others may not (e.g., to answer a proposal, a reply should succeed the proposal, it should occur within the time frame specified by the proposal, and no other reply should be deemed to have answered the proposal already—moreover, if the reply is an acceptance, the speaker and addressee of the reply should be the addressee and speaker of the proposal).

To support these conditions, we specify *shared proposals* and *shared replies* (below) as proposals and replies (as described in the *compositional* layer) that are shared (as indicated in Section 3.3). Using these notions, we defined that a proposal would be a *sound proposal* if its utterance is shared between speaker and addressee, and if its reply time frame falls beyond its time of utterance (which is a pragmatic requirement to indicate that all proposals could be answered in the future). Likewise, a reply would be a *sound reply* if it is shared, and if there is a preceding shared proposal that could be answered by this reply and has not been answered by any other reply yet.

$$SharedProposals : Interval \times \downarrow Agent \times \downarrow Agent \to \mathbb{P}\, Utterance$$

$$\begin{aligned}
&\forall interval : Interval;\ agent_1, agent_2 : \downarrow Agent \\
&\bullet\ SharedProposals(interval, agent_1, agent_2) = \\
&\quad \{u : SharedUtterances(agent_1, agent_2) \mid \\
&\qquad u.time \in interval.timeframe \land \\
&\qquad ToPropose(u.speechact) \neq \varnothing\}
\end{aligned}$$

$$SharedReplies : Interval \times \downarrow Agent \times \downarrow Agent \to \mathbb{P}\, Utterance$$

$$\begin{aligned}
&\forall interval : Interval;\ speaker, addressee : \downarrow Agent \\
&\bullet\ SharedReplies(interval, speaker, addressee) = \\
&\quad \{u : SharedUtterances(speaker, addressee) \mid \\
&\qquad (u.time \in interval.timeframe) \land \\
&\qquad (\exists reply : ToReply(u.speechact) \\
&\qquad\quad \bullet\ reply \notin Accept \lor \\
&\qquad\qquad (speaker = u.speechact.addressee \land addressee = u.speechact.speaker))\}
\end{aligned}$$

Therefore, that a propose token occurs between two agents at a given time is a *sound proposal* if there is a shared utterance with that token, and if this token has a reply time frame ending after the utterance.

$$SoundProposals : Time \times \downarrow Agent \times \downarrow Agent \rightarrow \mathbb{P} \downarrow Propose$$

$$\begin{aligned}
&\forall time : Time;\ agent_1, agent_2 : \downarrow Agent \\
&\bullet\ SoundProposals(time, agent_1, agent_2) = \\
&\quad \{propose : \downarrow Propose \mid \\
&\qquad (time \leq propose.reply.until) \wedge \\
&\qquad (\exists u : SharedProposals(at(time), agent_1, agent_2) \\
&\qquad\ \bullet\ propose \in ToPropose(u.speechact))\}
\end{aligned}$$

Likewise, that a reply token occurs between two agents at a given time is a *sound reply* (not shown) if there is a shared proposal that could be answered by the reply and has not been answered yet.[2]

Next, we define that an utterance is *proposing* (as shown below) if it has propose tokens that are proposals (as defined in the *compositional* level) and sound proposals (as defined above).

$$Proposing : Utterance \rightarrow \mathbb{P} \downarrow Propose$$

$$\begin{aligned}
&\forall u : Utterance \\
&\bullet\ Proposing(u) = \\
&\quad ToPropose(u.speechact) \cap \\
&\quad SoundProposals(u.time, u.speechact.speaker, u.speechact.addressee)
\end{aligned}$$

Similarly to the *compositional* layer, messages in this layer can be refined to create other message definitions, such as *accepting*, *rejecting*, *offering*, *requesting*, and so on. For example, an utterance is *requesting* (below) if it has tokens that are proposing and are requests.

$$Requesting : Utterance \rightarrow \mathbb{P} \downarrow Propose$$

$$\forall u : Utterance \bullet Requesting(u) = Proposing(u) \cap ToRequest(u.speechact)$$

4.3 The commitment state layer

The *commitment state* layer builds upon the *compositional* and *conversational* layers, and identifies messages based on the commitment states they attempt to advance when uttered. That is, messages are identified given their form but would only be successful if they fit the state of the commitments currently in place when they are uttered.

To support message definitions, we identify the commitment states that agents share at a given time. For example, *InactiveCommitments* and *RejectedCommitments* (below) identify those shared commitments (as defined in Section 3.3) that are inactive and rejected, respectively.

[2] This is specified by *proposed*, which maps each shared proposal that could be replied at a given time to the replies that could reply to it; and *replied*, which maps a subset of the proposals in *proposed* with one of its replies, where each reply replies to only one proposal. Thus, a reply is sound if it can reply to a non-replied proposal.

$$InactiveCommitments : Time \times \downarrow Agent \times \downarrow Agent \rightarrow \mathbb{P}\, SocialCommitment$$

$$\begin{aligned}
&\forall\, time : Time;\ a_1, a_2 : \downarrow Agent \\
&\bullet\ InactiveCommitments(time, a_1, a_2) = \\
&\quad \{sc : SocialCommitment \mid \\
&\qquad \exists\, state : SharedCommitments(time, a_1, a_2) \mid \\
&\qquad\quad first(state) = sc \\
&\qquad \bullet\ last(state) \in Inactive\}
\end{aligned}$$

$$RejectedCommitments : Time \times \downarrow Agent \times \downarrow Agent \rightarrow \mathbb{P}\, SocialCommitment$$

$$\begin{aligned}
&\forall\, time : Time;\ a_1, a_2 : \downarrow Agent \\
&\bullet\ RejectedCommitments(time, a_1, a_2) = \\
&\quad InactiveCommitments(time, a_1, a_2) \cup CancelledCommitments(time, a_1, a_2)
\end{aligned}$$

We use these definitions to specify messages such as proposals to adopt new commitments, proposals to discharge accepted commitments, acceptances to adopt commitments, and so on. To illustrate how these messages are defined, we present below a proposal to adopt a commitment, which is then refined into a request message.

$$ProposingAdoption : Utterance \rightarrow \mathbb{P} \downarrow Propose$$

$$\begin{aligned}
&\forall\, u : Utterance \\
&\bullet\ ProposingAdoption(u) = \\
&\quad \{p : Proposing(u) \mid \\
&\qquad p.proposing \in Add \wedge \\
&\qquad p.proposing.commitment \in InactiveCommitments(u.time, \\
&\qquad\qquad u.speechact.speaker, u.speechact.addressee)\}
\end{aligned}$$

$$RequestingAdoption : Utterance \rightarrow \mathbb{P} \downarrow Propose$$

$$\begin{aligned}
&\forall\, u : Utterance \\
&\bullet\ RequestingAdoption(u) = ProposingAdoption(u) \cup Requesting(u)
\end{aligned}$$

4.4 The joint activity layer

The *joint activity* layer builds upon the *compositional*, *conversational* and *commitment state* layers, and identifies messages according to the commitment actions that commitments advance in joint activities.

We make use of the NetBill protocol [19], which is an eight-step protocol for buying electronic goods over the Internet, to exemplify an activity with actions, and the commitment messages bringing about these actions. As shown in Fig. 5, the protocol starts when a consumer requests a quote for a product. If the quote from the merchant (step 2) is accepted (step 3) then the merchant sends an encrypted version of the product to the customer (step 4), who sends an electronic payment order (EPO) to the merchant (step 5). This EPO is forwarded to a NetBill server (step 6), who will debit the consumer and credit the merchant. If successful, the server sends a receipt with a decrypting key to the merchant (step 7) for forwarding to the customer (step 8).

The protocol involves three agent roles (a customer, a merchant and a NetBill server) and several actions, each associated with a particular role (e.g., the customer creates an EPO, the merchant produces encrypted goods). It is important to recall

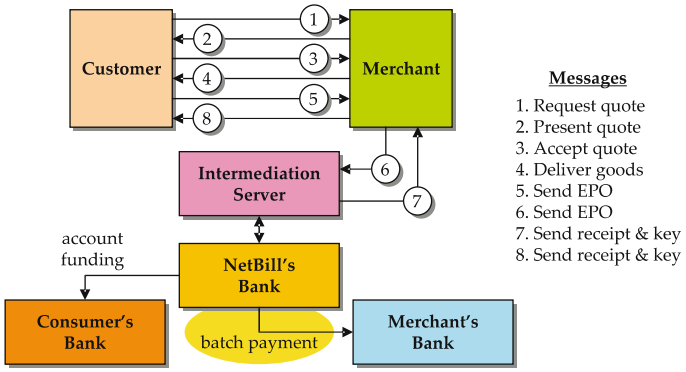

Fig. 5 The NetBill protocol

that we are not concerned with how these actions are performed but rather with the commitment messages that each of these roles can utter to bring about these actions.

Once actions, data and agent roles are defined, they are integrated and correlated in joint activities, such as the NetBill activity (below). This activity defines strict dependencies between data and actions, and also between actions and agent roles.

NetBill
JointActivity

$customer : \downarrow Customer$
$merchant : \downarrow Merchant$
$server : \downarrow NetBillServer$
$description : Description$
$quote : Quote$
$goods : Goods$
$epo : EPO$
$receipt : Receipt$
$quoting : ToQuote$
$delivering : ToDeliverGoods$
$paying : ToIssueEPO$
$confirming_1, confirming_2 : ToIssueReceipt$

$(customer = quoting.receiver = delivering.receiver = paying.sender = confirming_1.receiver) \wedge$
$(merchant = quoting.sender = delivering.sender = paying.receiver = confirming_1.sender = confirming_2.receiver) \wedge$
$(server = confirming_2.sender)$

$(description = quoting.description) \wedge$
$(quote = quoting.quote = delivering.quote) \wedge$
$(goods = delivering.goods = paying.goods) \wedge$
$(epo = paying.epo = confirming_1.epo = confirming_2.epo) \wedge$
$(receipt = confirming_1.receipt = confirming_2.receipt)$

For example, the *paying* action would have *goods* as its data input, the customer as the sender, and the merchant as the receiver of an EPO data output. That is, given the goods delivered (there is a *delivery* action with the *goods* as its data output), the customer will issue an EPO to the merchant. Just as data create dependencies between actions, roles cluster actions by identifying their performer. For example, the customer would be the receiver of a quote and the goods, the issuer of an EPO, and the recipient of a receipt.

5 Sequencing: A netbill interaction

Figure 6 shows the messages exchanged between customers and merchants in NetBill, and how these messages are modelled in the *pfp*.[3]

That the customer has requested a quote (message 1) indicates that agents share the commitment that the merchant will reply.[4] When the merchant presents a quote (message 2), he is not only accepting to provide a quote (labelled as commitment α) but he is both proposing to discharge α because he is informing a quote, and also proposing that the customer buy the goods based on this quote (labelled as commitment β). At this point, agents share commitment α (the merchant sends a quote), and two commitments in which the customer replies to the discharge of α and the adoption of β. When accepting the quote (message 3), the customer indicates her acceptance to discharge α and adopt β, which becomes the only shared commitment. Since β indicates that the merchant should deliver the goods, both agents know how the interaction should continue. Delivering the goods (message 4) is defined as a message in which the merchant proposes both to discharge delivering the goods (commitment β), since he is delivering them, and to adopt that the customer pays for them by issuing an EPO (commitment γ). This message results in new shared commitments where the customer replies to the discharge of β and the adoption of γ. When the customer sends an EPO (message 5), she accepts these operations, and also proposes to discharge issuing an EPO (commitment γ), given that she is issuing one, and to adopt that the merchant issues a receipt and a key (commitment δ). This message results in the shared state of commitment γ, and two commitments in which the merchant replies to the proposals to discharge γ and adopt δ. When the merchant sends a receipt and a key (message 8), he accepts to discharge γ and adopt δ, and proposes to discharge δ (sending a receipt and a key), given that he is sending these items. Message 9 (which is not mandated by NetBill but is required by the *pfp*) indicates the customer's acceptance to discharge δ. That there are no conversational commitments left at this point indicates the end of the interaction. This example illustrates the logical dependency between messages that result in structured conversations.

[3] Even though the figure shows that the *pfp* can be used to model the interaction defined in NetBill, the *pfp* can support many other message sequences that achieve the same outcome specified by NetBill. Examples of other interactions include merchants sending unrequested quotes (a case not that different from regular advertisement), or a customer requesting goods without asking first for a quote (which may happen in cases where price is known or unimportant to the customer). Readers are refered to [7] for an analysis of the flexibility afforded by the *pfp*.

[4] The policies applied after each *pfp* message are described in [7].

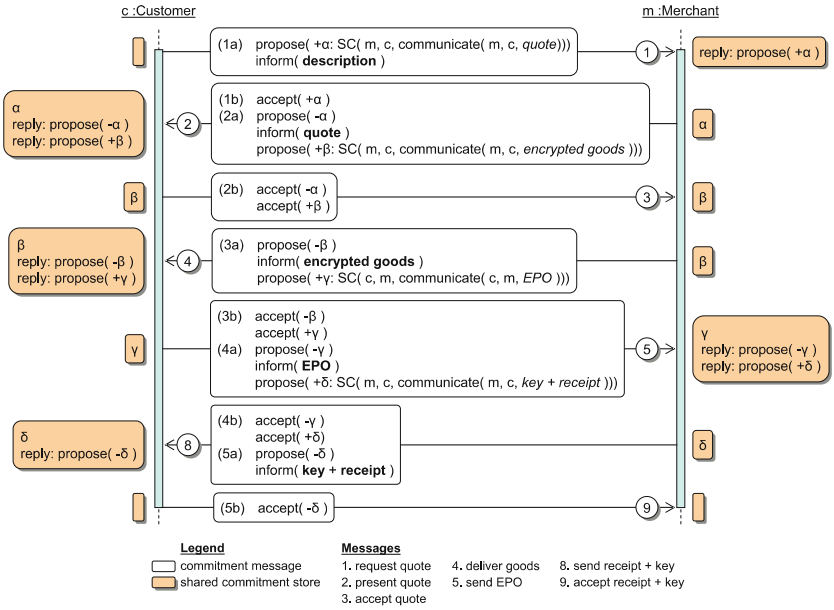

Fig. 6 Interaction between customer and merchant in the NetBill protocol

6 Meaning: form and success

Messages can be meaningful due to its form and success as transition vehicles between conversational, commitment and activity states. Whereas form depends on the characteristics of messages, success depends on the context in which messages happen. Even though form and success are not new as measurements of meaning, we explore whether they could apply to contexts that are solely based on the states created by the occurrence of commitment messages. In particular, we examine message form and success across the four layers presented in Section 4.

6.1 Form

We can use the message *RequestQuote* to exemplify how form is defined throughout the four layers in the model. This message, which is shown in Figs. 5 and 6 as message 1, can be informally described as a message from a customer to a merchant proposing to adopt a commitment in which the merchant is responsible for communicating a quote to the customer based on a description of goods.

Figure 7 shows in detail the compositional form of this message. The *compositional* layer describes it as: a proposal (*ToPropose*), since it seeks to modify the commitment store of agents; an adoption (*ToAdopt*), since it aims at adding a commitment; a request (*ToRequest*), since the speaker and addressee are the creditor and debtor of the commitment; and a quote (*ToQuote*), since it indicates an action in which a quote is communicated. The message *ToRequestQuote* puts these elements together, and also indicates that the addressee is the actor of producing and then communicating a quote to the speaker.

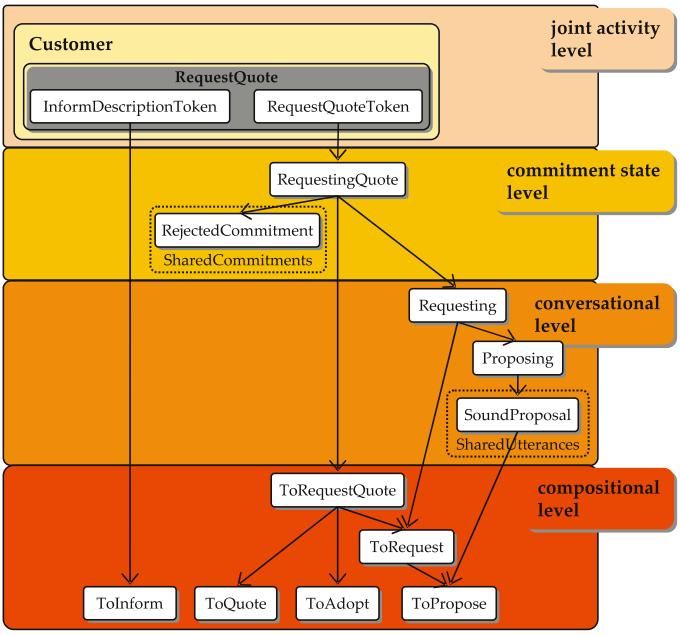

Fig. 7 NetBill *RequestQuote* message meaning lattice

The *conversational* layer specifies that once uttered, messages should become part of the shared utterances of agents. In this context, any utterance could be a proposal (*Proposing*) if it is a sound proposal (specified in Section 4.2), and could be a request (*Requesting*) if it has the properties of proposing and those of a request. Besides being a shared utterance, the only condition for a proposal to be proposing would be that its reply time allows room for a future answer.

The *commitment state* level indicates that an utterance requesting a quote (*RequestingQuote*) is meaningful only if the commitment being referred to is in a rejected state; that is, if it is not already recorded in the commitment store of agents as an adopted shared commitment.[5]

Lastly, the *joint activity* level indicates that requesting a quote (*RequestQuote*) is an utterance that simultaneously requests to uptake a non-adopted commitment to quote (*RequestQuoteToken*) and informs the description of the goods to be quoted (*InformDescriptionToken*). This message is related to the role of a customer, which indicates that any agent uttering it should have the profile to uptake this role.[6]

The *Customer* agent role, which is partially defined below, shows the definition of *RequestQuote*, *RequestQuoteToken* and *InformDescriptionToken*, where the two latter define the characteristics of utterances requesting to adopt a commitment to

[5] This includes not only the case in which the commitment is active but also when it is in a fulfilled or violated state. Being able to automatically retract a fulfilled or violated commitment (rather than to wait for an explicit discharge) should apply only in special circumstances, e.g., when sanctions and rewards do not apply [14].

[6] Whether an agent could interact with others in a certain role would depend on whether this agent is able, willing and authorized to do so. An interesting approach to authorizations, roles and commitments is explored in [9].

quote, and informing the description of goods, respectively. The function *Requesting-Quote* (not shown), which is used in *RequestQuoteToken*, defines the features that utterances requesting to adopt a commitment to quote should have.

The aggregated meaning of *RequestQuote* can be summarized as that of a message spoken by a customer in which she informs a description of goods, and requests that this description be used by the addressee as the basis for informing her of a quote (that the merchant is the addressee is established in the *NetBill* joint activity in Section 4.4).

Customer

$\uparrow(RequestQuote)$

Agent

RequestQuoteToken

$quoting? : ToQuote$

$utterance! : Utterance$

$utterance!.speechact.speaker = quoting?.receiver = self \land$

$utterance!.speechact.addressee = quoting?.sender$

$\exists propose : RequestingQuote(utterance!, quoting?)$

$\bullet\ propose \in utterance!.speechact.tokens$

InformDescriptionToken

$description? : Description$

$utterance! : Utterance$

$\exists inform : ToInform(utterance!.speechact)$

$\bullet\ description? \in inform.informing$

$$RequestQuote \mathrel{\widehat{=}} [\, quoting? : ToQuote;\ description? : Description \,] \mid quoting?.description = description? \,] \bullet RequestQuoteToken \land InformDescriptionToken$$

6.2 Success

Pre-conditions indicate a set of conditions whose fulfillment would make the utterance of a message succeed.

We identify pre-conditions using the state of the activities, commitments and conversations shared by agents. It is advantageous to define pre-conditions this way because states depend on utterances, which are observable events. Since the state of activities is advanced by the state of commitments, which is advanced by the state of conversations, which is advanced by the messages uttered by agents, any pre-conditions defined in those terms could be directly verifiable by any agent that has kept track of these events. As a result, any agent should be able to assert whether a message could be uttered or not given its pre-conditions.

In addition, since we rely on the assumption that communications are reliable, and that agents follow the same agreement and commitment conventions, then agents should have identical representations of shared states given that they have witnessed the same utterances. If that was not the case, discrepancies could exist between what agents believe they should be sharing. Interestingly, if any discrepancies were

identified (e.g., an utterance would not succeed given its pre-conditions, commitments are not fulfilled when expected), agents could resort to additional interactions to clarify these exceptions (future work).

Pre-conditions could or could not be specified for any particular message. For example, a message requesting a quote (*RequestQuote*) could be issued at any time as long as there is a description that could be used as the input for quoting. In contrast, other messages should only be uttered at certain times based on the current state of interactions. For example, presenting a quote (*PresentQuote*), shown in Figs. 5 and 6 as message 2, should only be uttered if a customer has requested a quote, and if the reply to such proposal could be issued at that time.[7]

The *PresentQuote* message in the *Merchant* role is shown below.

Merchant

$\lceil(PresentQuote)$
Agent
$PresentQuote \mathrel{\widehat{=}}$
$[\, quote? : Quote;\ quoting? : ToQuote;\ delivering? : ToDeliverGoods \mid$
$(quote? = quoting?.quote = delivering?.quote) \land$
$(\exists\, reply : RepliesToQuoting(now, self, quoting?)$
$\bullet\ \exists\, speak : ToSpeak \mid speak = reply.replying.commitment.action$
$\bullet\ now \in speak.performance.timeframe)\,]$
$\bullet\ RequestQuoteDischargeToken \land OfferGoodsToken \land InformQuoteToken$

This message is composed of the operations *RequestQuoteDischargeToken*, *OfferGoodsToken* and *InformQuoteToken* (none of which are shown). These messages define the characteristics of a request to discharge quoting, an offer to adopt delivering goods, and an inform of a quote, respectively. The remainder of the definition (shown between square brackets) indicates the pre-conditions of this message. The parameters in *PresentQuote* indicate that the quote being informed, the quote in the quoting action, and the quote upon which the delivery of goods is based, are the same. Lastly, *RepliesToQuoting* (not shown) supports identifying whether a commitment to reply to an adoption to quote exists, and whether this reply could be issued now.

It is worth highlighting that by using public criteria, such as shared utterances and shared commitments, and common agreement conventions to modify commitment stores, the form and success of messages can be justified and verified in open systems. If any discrepancies were identified (e.g., proposing to discharge a non-accepted commitment, replying to a non-existing proposal), agents could resort to additional interactions where these exceptions could be clarified. The type and form of these recovering dialogues is yet to be explored.

[7] This view does not preclude that merchants could provide unrequested quotes. Even though the form of requested and unrequested quote messages may likely be the same, these messages could be defined in different roles, e.g., a door-by-door salesperson (who issues unrequested quotes) versus a store salesperson (who issues requested quotes only). That an agent has received an unrequested quote would signal the role the speaker is portraying (e.g., "overzealous salesman") or may indicate discrepancies in shared states (e.g., "I mistook you by the person that requested me a quote.", "I forgot I asked you for a quote earlier today.").

7 Related work

Conversations and commitments have been the subject of previous studies. Some have studied commitments in argumentation [21], where the evolution of conversations is motivated by the commitments that are implied (and not necessarily made explicit) in dialogs (e.g., [13, 16]), while others have studied the mechanics of conversations using commitment-based conventions. We share these views, and aim to identify the public elements guiding conversations.

Other efforts pursuing this goal include those led by Chaib-draa [2] (who specifies dialog games to advance commitment states), Colombetti [9] (who specifies message meaning based on commitment states and operations) and Singh [12] (who specifies an algebra to reason about commitment protocols). These efforts share many similarities, and their differences have more to do with focus, abstraction and formalization than with fundamental principles.

In [12], Mallya and Singh present an algebra for reasoning about protocols. Its strength resides on the simplicity with which complex protocols can be composed and analyzed, which contrasts with the complexities of defining even the simplest conversations in our model. This is explained by their high and our low level of abstraction. At the low end, we use action commitments as the strict logical connection between states and messages, thus indicating why conversations are structured in a certain way. Once the structure of conversations is understood, we can abstract from details of agreement conventions and lengthy message definitions, and start thinking in terms of labeled states and messages, that is, in terms of protocols. It is at this higher level of abstraction where the use of an algebra becomes an asset.

In [9], Fornara, Vigano and Colombetti present message definitions in the context of an institution, along with the norms and authorized behaviour that agents in certain institutional roles should follow. This conceptualization is akin to that of joint activities, although it is enriched with the concept of authorizations and norms. Interestingly, the conjunction of authorizations and the definition of messages as declarations allows utterances to affect the state of commitments without an explicit agreement. That is, by accepting to participate in an activity, agents surrender part of their autonomy and agree that certain messages could immediately affect their commitment stores (note that whether an acceptance could or could not be redundant in such cases should be based on the aggregated meaning of messages [11]). Extending our model to account for institutions is part of our future work. One aspect that is still missing from this approach is how conversations should evolve, that is, any formal indication of which agent is to utter the next message (which is different than indicating the messages that are allowed to be uttered) at each institutional state.

In [10], Kagal and Finin, propose a framework to support decision making in conversations. Their premise is that conversation protocols can be complemented with conversation policies defined in terms of permissions (authorizations) and obligations (responsibilities), which would influence the choice of messages uttered at any conversational state. In addition, the framework makes provision for meta-policies, that is, policies that would resolve conflicts among contradicting policies. To some extent, this approach shares the same view of authorizations and responsibilities as presented by Fornara et al. [9], but without the benefit of institutions and roles as their context of application. Nevertheless, Kagal and Finin explore meta-policies, which is a notion still to be included into Fornara, et al.'s work.

8 Conclusions & future work

In his influential study of language use [3], Clark identifies utterances as signals with two complementary types of meaning: *speaker's meaning*, which is defined in terms of their use to communicate intent, and *signal meaning*, which is defined in terms of their use as coordinating devices advancing joint activities. Although subtle, their difference is striking: whereas speaker's meaning appeals to the reasons behind advancing an activity, signal meaning puts forth a public token advancing it.

Traditionally, the semantics of agent communication languages have emphasized speaker's meaning, as reflected by the use of speech acts and mental states in commonplace message definitions. This approach is unrivaled for communicating intent, since agents can readily know the intention of messages by just observing their definitions rather than by inferring it from the context of interaction. Yet, definitions are given independently of any conversational context, and their use in open systems mandates the sincerity of agents. Signal meaning, on the other hand, has been kept as a low profile component of meaning and is not addressed by any standardizing effort.

In this paper, we present our initial efforts towards merging signal meaning and conversational structure using a social commitment-based approach. We argue that messages can be meaningful independently of the intentions behind their utterance if they advance the state of the commitments that exist in the context in which they happen. This work fits within our long-term goal of supporting agent interactions in open environments. This goal requires identifying theoretical principles that can be used to define a practical framework upon which interactions can be built. As a result, the primary contribution of our research is to advance the state-of-the-art in agent communication languages by proposing an operational model that combines message meaning and conversational structure in one comprehensive approach.

Future work includes exploring more elaborated commitment state transitions, which is an effort already underway within the context of sanctions [14], and expanding the model to cover speaker meaning [8] and electronic institutions. On the practical side, we are planning to implement the model in a computer system, which is a step that would allow us to validate and refine it against a larger number of examples.

Acknowledgements We are grateful for the support received from the National Science and Engineering Research Council (NSERC) of Canada. We extend our gratitude to the anonymous reviewers, who provided invaluable suggestions to improve our work.

Appendix

Object-Z overview

Object-Z [20] is an object-oriented extension of Z [5], which is a formal specification language based on first-order logic and set theory. Object-Z extends Z by supporting classes, inheritance and polymorphism. In brief, definitions in Z are organized around functions and schemas. Schemas are extended in Object-Z to specify classes and methods.

A function is exemplified below. This function, named *multiply*, indicates that for any pair of real numbers given as arguments, the function results in a real number equal to the multiplication of the arguments.

$$multiply : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$$

$$\forall number_1, number_2 : \mathbb{R} \bullet multiply(number_1, number_2) = number_1 * number_2$$

Schemas support the specification of states and transitions, which are essential to define classes and methods. The class *CounterClass* is shown below as an example. This class defines two methods, one with no name, and the other one named *increment*. The method without a name defines the class' fields (in this case, a natural number named *counter*). The *increment* method specifies that the value of the field *counter* is incremented by one each time that this method is invoked (*counter'* indicates the state of *counter* after the execution of the method).

CounterClass

$$counter : \mathbb{N}$$

increment

$$\Delta(counter)$$

$$counter' = counter + 1$$

The class *MultiplyClass* (shown below) subclasses *CounterClass* and defines the methods *multiply* and *multiplication* (where only the latter is defined as public, as indicated immediately below the class name). The method *multiply* takes the real numbers $number_1?$ and $number_2?$ as arguments and returns a real number *result*! (where *?* and *!* indicate input and output variables, respectively). This method specifies that the value of *result*! is equal to the multiplication of $number_1?$ and $number_2?$ Lastly, the method *multiplication* is defined as the concurrent invocation of *multiply* and *increment* (inherited from *CounterClass*) thus counting the number of times that multiplications are performed.

MultiplyClass

$$\upharpoonright (multiplication)$$

CounterClass

multiply

$$number_1?, number_2?, result! : \mathbb{R}$$

$$result! = number_1? * number_2?$$

$$multiplication \mathrel{\widehat=} multiply \wedge increment$$

References

1. Castelfranchi, C. (1995). Commitments: From individual intentions to groups and organizations. In *Proceedings of the first international conference on multi-agent Systems*, June 1995, (pp. 41–48), San Francisco, CA.
2. Chaib-draa, B., Labrie, M-A., Bergeron, M., & Pasquier, P. (2006) DIAGAL: An agent communication language based on dialogue games and sustained by social commitments. *Autonomous Agents and Multi-Agent Systems*, *13*(1), 61–95.
3. Clark, H. H. (1996). *Using language*. Cambridge University Press.

4. Craig, R. T., & Tracy, K. (1983). *Conversational coherence: Form, structure, and strategy*. Sage Publications.
5. Diller, A. (1990). *Z: An introduction to formal methods*. Sussex, England: John Wiley & Sons, Inc.
6. Flores, R. A., & Kremer, R. C. (2003). To commit or not to commit: Modelling agent conversations for action. *Computational Intelligence, 18*(2), 120–173.
7. Flores, R. A., & Kremer, R. C. (2004). A principled modular approach to construct flexible conversation protocols. In A. Y. Tawfik, & S. D. Goodwin (Eds.), *Advances in Artificial Intelligence: Canadian AI 2004*, *Lecture Notes in Computer Science* (Vol. 3060, pp. 1–15). May 2004. Springer Verlag.
8. Flores, R. A., Pasquier, P., & Chaib-draa, B. (2006). What do agents commit to do when they commit to something? *Work in progress*.
9. Fornara, N., Vigano, F., & Colombetti, M. Agent communication and institutional reality. *Autonomous Agents and Multi-Agent Systems*, forthcoming.
10. Kagal, L., & Finin, T. Modeling communicative behavior using permissions and obligations. *Autonomous Agents and Multi-Agent Systems*, forthcoming.
11. Kremer, R. C., & Flores, R. A. (2005). Using a performative subsumption lattice to support commitment-based conversations. In *Proceedings of the fourth international joint conference on autonomous agents and multiagent systems*, July 2005, Utrecht, The Netherlands.
12. Mallya, A. U., & Singh, M. P. A semantic approach for designing commitment protocols. *Autonomous Agents and Multi-Agent Systems*, forthcoming.
13. Parsons, S., McBurney, P., & Wooldridge, M. J. (2004). The mechanics of some formal inter-agent dialogues. In F. Dignum (Ed.), *Advances in agent communication*, *Lecture Notes in Artificial Intelligence* (Vol. 2922, pp. 329–348). Springer Verlag.
14. Pasquier, P., Flores, R. A., & Chaib-draa, B. (2004). Modelling flexible social commitments and their enforcement. In M. P. Gleizes, A. Omicini, & F. Zambonelli (Eds.), *Fifth international workshop on engineering societies in the agents world*, *Lecture Notes in Artificial Intelligence* (Vol. 3451, pp. 153–165) Tolouse, France, October 2004. Springer Verlag.
15. Pitt, J., & Mamdani, A. (1999). Some remarks on the semantics of FIPAs agent communication language. *Autonomous Agents and Multi-Agent Systems, 2*(4), 333–356.
16. Reed, C. A. (1998). Dialogue frames in agent communication. In *Proceedings of the third international conference on multiagent systems (ICMAS 98)* (pp. 246–253). Paris, France: IEEE Press.
17. Singh, M. P. (1991). Social and psychological commitments in multiagent systems. In *AAAI fall symposium on knowledge and action at social and organizational levels*, November 1991, Monterey, California.
18. Singh, M. P. (1998). Agent communicational languages: Rethinking the principles. *IEEE Computer, 31*(12), 40–47.
19. Sirbu, M. (1997). Credits and debits on the Internet. *IEEE Spectrum, 34*(2), 23–29.
20. Smith, G. (2000). *The object-Z specification language*. Kluwer Publishers.
21. Walton, D. N., & Krabbe, E. C. W. (1995). *Commitment in dialogue: Basic concepts of interpersonal reasoning*. State University of New York Press.