A Formal Framework for Agent Interaction Semantics

Shamimabi Paurobally
University of Southampton
Southampton SO17 1BJ, UK.
sp@ecs.soton.ac.uk

Jim Cunningham
Imperial College
London SW7 2BZ, UK.
rjc@doc.ic.ac.uk

Nicholas R. Jennings
University of Southampton
Southampton SO17 1BJ, UK.
nrj@ecs.soton.ac.uk

ABSTRACT

Although informative, the semantic definition proposed for the most standard agent communication language (FIPA ACL 1997) is complicated and contentious, while published interaction protocols (IPs) tend to be ambiguous, incomplete, and unverified with respect to message semantics. To clarify and help rectify these problems, this paper proposes an integrated framework based on Propositional Dynamic Logic and Belief and Intention modalities (called the PDL-BI language). Specifically, we provide an axiomatisation of PDL-BI and for an agent's propositional attitudes (beliefs and intentions) and social attitudes (such as sincerity and trustworthiness). Then, we suggest a revised and simpler core semantics for many of the FIPA ACL speech acts, which, in turn, lead to the specification of the semantics of IPs. As a case study, we specify the semantics of the contract net protocol (CNP) in PDL-BI, which allows to prove that the CNP terminates.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed AI

General Terms

Languages, Verification, Theory

Keywords

semantics, belief, intention, ACL, protocol, interaction state

1. INTRODUCTION

Social interactions, such as cooperation, coordination and negotiation, are enacted through a variety of agent communication languages (ACLs) and interaction protocols (IPs). An ACL (for example KQML, FIPA ACL [1]) specifies the individual communicative acts (CAs), typically as classes of asynchronous messages modelled on the theories of speech acts [10] . The 1997 semantic specification for FIPA ACL is expressed using a logic of belief and intention, derived from [9]. While this specification is informative, it has been criticised on various grounds [8],[12], not least that it is unverifiable. These criticisms can be contested. That is not our purpose here. An interaction protocol like the contract net protocol (CNP) [11], or an English auction protocol, purports to specify the message sequences that can lead towards a goal state. Ideally, in an agent world, each message would be the content conveyed by an individual communicative act of the ACL. However, the published specifications of protocols suffer from ambiguities and incompleteness [7]. This lack of precision arises from the inherent inexpressiveness of a diagrammatic representation such as Petri-nets and AUML [6], and from a focus on a conceptual level of discussion using informal language rather than formal logic. Consequently, while the published protocol specifications may well be helpful for comprehension, in practice they are inadequate for prescribing the individual messages of an implementation.

Against this background, we propose a framework that separates the different components for developing agent interactions. To specify these components, we first specify the PDL-BI language, which combines an extended form of Propositional Dynamic Logic (PDL) with Belief and Intention (BI) modalities. We provide an axiomatisation of PDL-BI for representing and reasoning about the semantics of ACLs and IPs, and we formulate axioms for expressing both an agent's social attitudes (such as sincerity or cooperation) and its belief about the exchange of messages over a network. Given this framework, we specify a simpler semantics for an ACL and provide a similar style of semantics for the contract net protocol (CNP) when expressed in the ACL. Together, these allow us to prove the properties of the CNP. Using the CNP as a case study serves as an example of how to derive the semantics of IPs and verify IPs.

This paper advances the state of the art by addressing a number of unresolved issues in agent interactions. *First*, we provide a layered framework so as to separate treatment of message delivery, sincerity, belief intention axioms, and implicit protocol issues within the existing style of FIPA ACL logic. This separation of concerns enables us to simplify verification of an IP with respect to the message semantics. *Second*, is the specification of PDL-BI for expressing the semantics of ACLs and IPs. Here, a belief logic is a useful ideal for giving epistemic status to the consistent, but not necessarily true, internal propositions that can be used by a designer to express and reason about information internal to an agent. Treating intention in a similar way is also a useful ideal for succinct reasoning about a goal state without getting into more temporal reasoning. The durability of the Belief-Desire-Intention paradigm for practical deliberative agents also provides a heuristic justification for this sort of reasoning. For these reasons, PDL-BI is used as a common language for expressing the semantics of ACLs and IPs. This relation be-

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

tween the semantics of ACLs and IPs, although implicit in current research in IPs, has never been made explicit before. In light of criticisms about ambiguities in ACL semantics and indeed in our earlier endeavours to address these issues, a *third* contribution is to provide a simpler, corrected and clearer core ACL semantics based on FIPA ACL rather than considering a more radical change in style (say to commitment-based) or in abstraction level e.g. in institutional signalling conventions [4] or argumentation.

The remainder of this paper is structured in the following way. Section 2 provides an informal summary of the current limitations in ACLs and IPs. Section 3 presents the layers in our framework for facilitating agent interactions. In section 4, we specify the syntax and semantics of PDL-BI and provide axioms for its modalities and for expressing an agent's social attitudes. Section 5 presents our revised FIPA ACL semantics. These are used in section 6 to specify the semantics of the CNP and to prove the termination of its interpretation. Section 8 concludes.

2. ACLS AND IPS

Agent interactions, achieved through ACLs and IPs, can help towards interoperation in multi-agent systems. Through an ACL, such as FIPA ACL, an agent can communicate its intentions to other agents. In FIPA ACL there is a feasibility pre-condition (FP) and a rational effect (RE) associated with each CA. The former conjoins a sincerity condition (SC) with a typically more complex Gricean condition (GC) to preclude a redundant message. The RE expresses the condition that the sender may use in planning the communicative act. For example, consider the FIPA *inform* (or KQML *tell*), which is the basic CA. As a message it has parameters for sender s, receiver r, and propositional content ϕ. The SC is $B_s\ \phi$ and the RE is $B_r\ \phi$. The GC expresses the belief that the sender does not believe the receiver believes ϕ, or is uncertain about it.

The obvious criticism is that the Gricean condition introduces inessential complexity, even if the term "inform" is inappropriate without it. To remedy this, in this paper, we will simply drop the GC part of the FP (on the grounds that it is really a social protocol for human communication which need not be presumed in the context of any other interaction protocol), but retain the SC part as a simpler FIPA-like pre-condition, and allow the FIPA RE as a trustworthiness assumption (see section 4.5). Now, it turns out that the remaining FIPA ACL messages that we use can be re-expressed as special cases of the *inform* act in context. This was known when the standard was prepared, but, in some cases, was obscured and made erroneous by the inessential complexity we have sought to remove. For example a *propose* message, sent by a potential CNP contractor to the manager, is an *inform* with the propositional content being that *if* the sender believes that the receiver intends the action to be done by the sender, *then* the sender (will) intend this (too). So a sincere *accept* or *reject* requires belief by the manager in the propositional content of such a proposal, and becomes an *inform* message with content expressing the manager's intention, so that the contractor can discharge the conditional "promise".

There are other deeper issues with the FIPA semantics. The semantic conditions as expressed are sender oriented and there is no overt association between the semantic conditions and the occurrence of the message itself (after all, sincerity and non-redundancy are social, not mechanical conditions). Although these concerns have been pointed out in [12], we take a new step in removing much of their impact by re-expressing the semantics in the belief-intention logic itself. The key step is not obvious and indeed exploits an obvious "hack" in the logic, which is to avoid the detailed expression of temporality or causation by using the special formula $done(a, act)$, for any agent a and action act. In PDL-BI notation, we express $done(a, act)$ as $done(a.act)$. Now the formulae $I_s\ done(s.m)$ and $B_r\ done(s.m)$, respectively, express that the sender agent s intends that m be sent, and receiver agent r believes that m has been sent by s. These are the tightest pre- and post-conditions we can express in the logic. More details are given in section 4.4.

Given these problems in the semantics of ACLs, there are also weaknesses in the current representations of IPs. Ambiguities and incompleteness in specifications of several known protocols have been exposed in [7]. In this paper, we use the CNP as a case study because it involves multiple agents interacting in an open environment, and this multi-lateral feature reflects the limitations of current proposals to represent IPs [7]. Specifically, current representations in Petri Nets or AUML fail to concisely capture the binding between specific agents and exchanged messages. For example, the manager should not send an acceptance of contractor c_1's proposal to contractor c_2. Petri nets and the AUML notation show an explosion in diagrammatic complexity when representing interactions between multiple agents because the messages cannot be bound to a particular agent, without introducing new partitions or timelines, respectively.

3. FRAMEWORK FOR INTERACTION

Figure 1 illustrates the five layers in our proposed framework for specifying and verifying IPs. Specifically, PDL-BI is a multi-modal language extending PDL and including belief and intention modalities. Given this language, in the first (lowest) layer in figure 1 we provide axioms for the propositional attitudes of the agents (beliefs and intentions) in order to assist in the proof of properties. The second component (Communication Axioms) of the first layer consists of axioms regarding the beliefs and intentions of a sender and receiver about the message exchange between them. Here we assume that the agent interaction occurs above the standard Internet protocols and that sent messages are eventually relayed. An agent's propositional attitudes about message exchange are not explicit in current ACL proposals. Nevertheless, such relations are important because, for example, the sender must intend to send a message before actually sending it and the receiver must receive it before believing its RE.

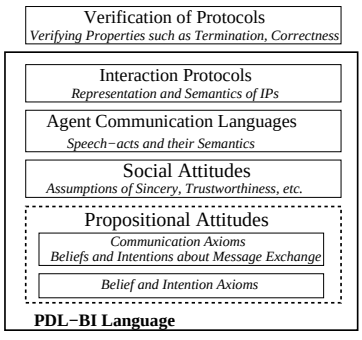

Figure 1: Framework for Facilitating IP Verification

The second layer in figure 1 specifies the assumptions made about the social attitudes of an agent. For example, these assumptions can be the sincerity and trustworthiness of the interacting agents or that sent messages are not repeated. The advantage of this layer is that it specifies and re-groups such social aspects of the agents instead of adding them to the semantics of each CA (as per the Gricean conditions in FIPA ACL). Although, for simplicity's sake, we assume sincerity and trustworthiness of the agents (section 4.5) in the specification of the semantics of the CAs in section 5, these assumptions can be relaxed in non-cooperative environments. The third layer consists of the definition of a set of CAs and their semantics. In layer 4, interaction protocols specify how individual CAs can be sequenced so as to form a meaningful conversation between agents towards a desired goal. There are two aspects to specifying an interaction protocol, the definition of the protocol and its semantics. An interaction protocol can be represented in AUML, Petri nets, statecharts or PDL while the semantics of the IP specify the meaning of a conversation and can be derived from the semantics of its constituent CAs.

Finally, these underlying layers facilitate verification of the properties of an IP (such as correctness, completeness and termination). Such properties are important for successful agent conversations and for choosing between various IPs. For example, an incorrect or incomplete protocol can allow contradictory actions (such as a manager accepting and rejecting the same proposal) or it may allow misunderstandings between participants (such as a contractor not sending back the results of the execution of its proposal). There are a number of methods that can be used for verifying IPs (e.g. model checking or theorem proving). The remainder of this paper is concerned with specifying the components of each layer and applying them in a case study.

4. THE PDL-BI LANGUAGE

4.1 Syntax

The PDL-BI language is an extension of PDL and includes modal operators for belief and intention similar to BDI logics. Monadic modal operators for a belief (B_i) and an intention (I_i) of an agent i are used as intellectual props for deductive inference and treated as independent except for explicit interaction axioms. The syntax of PDL-BI is defined in table 1. We assume throughout that each atomic formula, agent and instance of an atomic process can be denoted by a distinct identifying term.

Let *atomic formulae* be denoted by p, q, ... and let ϕ denote the set of atomic formulae. Examples of an atomic formula are *proposed, informed* and *rejected*. Let φ denote the set of formulae formed from atomic formulae using the standard connectives ($\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$), the modal operators and the last two formulae in table 1. Let A denote a formula in φ. The formula $\bot =_{def} (p \wedge \neg p)$ holds. Let G denote a *finite group of agents* (for example, a group of contractors), i, j an agent, and $\mathcal{A}g$ an agent or a group of agents in G. The syntax of PDL-BI remains propositional because we only deal with finite sets, and for convenience we use quantification over such finite sets. Let ϖ denote an *atomic process*. Examples of atomic processes are *propose*, *accept* and *inform*. We assume atomic processes are successfully executed. As for PDL, complex processes γ are generated from the set of atomic processes [2].

Table 1: Propositional Syntax of PDL-BI

Formulae:	$A ::= p \mid \bot \mid \neg A \mid A_1 \rightarrow A_2 \mid A_1 \wedge A_2 \mid A_1 \vee A_2 \mid$
	$A_1 \leftrightarrow A_2 \mid [\gamma]A \mid B_{\mathcal{A}g}A \mid I_{\mathcal{A}g}A \mid p(\mathcal{A}g) \mid \gamma_1 :: \gamma_2$
Processes:	$\gamma ::= \varpi \mid \gamma_1;\gamma_2 \mid \gamma_1 \cup \gamma_2 \mid \gamma^* \mid A? \mid null$
	$\mid abort \mid \mathcal{A}g\ .\varpi$

Table 2: Semantics of PDL-BI

$M, w \models p$	iff $w \in V(p), p \in PROP$
$M, w \models [\gamma]A$	iff $\forall w_1 (wR_\gamma w_1$ implies $M, w_1 \models A)$
$M, w \models p(\mathcal{A}g)$	iff $M, w \models p$
$M, w \models (\gamma_1 :: \gamma_2)$	iff $M, w \models R_{\gamma_1} \subseteq R_{\gamma_2}$
$M, w \models (\mathcal{A}g.\varpi)$	iff $M, w \models R_{\mathcal{A}g.\varpi} \subseteq R_\gamma$
$R_{\gamma_1;\gamma_2} = R_{\gamma_1} \circ R_{\gamma_2}$	
$R_{\gamma_1 \cup \gamma_2} = R_{\gamma_1} \cup R_{\gamma_2}$	
$R_{\gamma*} = R^*_\gamma$	
$M, w \models B_{\mathcal{A}g}A$	iff $\forall w_1 \in W (wR_{B_{\mathcal{A}g}} w_1$ implies that $M, w_1 \models A)$
$M, w \models Ii_{\mathcal{A}g}A$	iff $\forall w_1 \in W (wR_{Ii_{\mathcal{A}g}} w_1$ implies that $M, w_1 \models A)$

The formula $[\gamma]A$ has the intended meaning: A holds after executing process γ. The complex process $(\gamma_1;\gamma_2)$ denotes the sub-process γ_1 followed by γ_2, the process $(\gamma_1 \cup \gamma_2)$ is either γ_1 or γ_2 non-deterministically, γ^* denotes zero or more iterations of process γ. A state test operator "?" allows sequential composition to follow only if the tested state holds. A *null* process represents no execution, while an *abort* process results in a failed state. We extend the program logic of PDL so as to express multi-agent interactions. Set notation is used to manipulate sets of agents. Our extensions are formulae of type $p(\mathcal{A}g)$ and $\gamma_1 :: \gamma_2$, and processes of type $\mathcal{A}g.\varpi$, and their semantics are given in the next section.

4.2 Semantics

The PDL-BI logic is multi-modal. Despite the need for Intention to be treated as a non-normal modality, the logic can be given a formal "possible worlds" semantics using an extended multi-relational Kriple model denoted by
$M = (W, R_\gamma, V, R_{B_{\mathcal{A}g}}, R_{I1_{\mathcal{A}g}}, R_{I2_{\mathcal{A}g}}, R_{I3_{\mathcal{A}g}})$ [2]. Here W is a non-empty set of worlds. The function V is an assignment from sets of possible worlds to propositions. $V : \phi \rightarrow 2^W$ where $V(p)$ is the set of worlds where atomic formula p holds. R_γ, $R_{B_{\mathcal{A}g}}$ and each $R_{Ii_{\mathcal{A}g}}$ are accessibility relations on the worlds in W (in order to model processes, beliefs, and intentions, respectively). The model semantics are then as specified in table 2, subject to the supplementary requirements that the relation $R_{B_{\mathcal{A}g}}$ is serial, transitive, and Euclidean (see [5]), and that each formula of the form $I_{\mathcal{A}g}A$ is translated into $\neg I1_{\mathcal{A}g} \neg (I2_{\mathcal{A}g}A \wedge \neg I3_{\mathcal{A}g}A)$ (see [3]). As we do not use the semantics of table 2 in any explicit way, it serves only as a statement that a fairly conventional, if complex semantics can be provided. We do, however, reason informally about multi-modal formulae, and recognize that a complex semantic model can have hidden problems.

The semantics of the standard PDL operators can be found in [2], here we explain the semantics of the operators we have added. The semantics of $M, w \models (\gamma_1 :: \gamma_2)$ states that all the worlds obtained through execution of process γ_1 are elements of the set of worlds possible through performing γ_2. For example, *EbayAuction::EnglishAuction* means that all rules in the English auction apply to the Ebay auction. The formula $p(\mathcal{A}g)$ such as *proposed(C)* holds in world w if and only if p holds in w. We treat an agent or a group of agents as

executing atomic actions through the expression $\mathcal{A}g.\varpi$ (e.g. $c_i.propose$ means c_i executes the *propose* process)[1]. Using set notation, we can express a joint process between two parties, for example $\{c_i, c_j\}.deliberate$.

4.3 Axiomatisation

In this sub-section, we define axioms holding in PDL-BI, as part of the first layer in figure 1. PDL-BI contains the axioms of PDL in addition to the KD45 axioms for belief. Axioms K, D, 4 and 5 respectively express closure under implication, consistency, and positive and negative introspection. We assume that each agent in a group has such a system of belief. The formula $E_G A$ is read as everyone in a group of agents, G, believes A, where A itself may express an agent's beliefs and intentions. We also re-use the FIPA SL *done* operator [1]. Here, *done(i.γ)* (*done(i,γ)* in FIPA SL) means that agent i has performed action γ. Pre-conditions p for doing γ can be expressed as part of a process (e.g. $p?;\gamma$). In addition to the KD45 axioms for belief, table 3 gives axioms for intentions and for beliefs about intentions. Let an arbitrary formula be denoted by A and a process by γ.

Table 3: Belief-Intention Axioms

(A0)	$p(\mathcal{A}g) \rightarrow p$
(A1)	$B_i B_i A \leftrightarrow B_i A$
(D-Int)	$I_i A \rightarrow \neg I_i \neg A$
(Int-A)	$B_i I_i A \leftrightarrow I_i A$
(Int-B)	$B_i \neg I_i A \leftrightarrow \neg I_i A$
(A2)	$I_i A \rightarrow I_i B_i A$
(A3)	$I_i A \rightarrow \neg B_i A$
(A4)	$B_i B_i A \rightarrow \neg I_i B_i A$
(A5)	$B_i A \rightarrow \neg I_i A$
(A6)	$I_i A \rightarrow \neg I_i I_i A$
(A7)	$B_i I_i \ldots B_i I_i A \leftrightarrow I_i \ldots I_i A$
(A8)	$I_i A \wedge B_i[\gamma]A \rightarrow [\gamma]B_i A$
(A9)	$I_i done(\gamma) \wedge B_i[\gamma]A \rightarrow [\gamma]B_i A$
(A10)	$I_i A \wedge B_i[\gamma]A \rightarrow I_i done(\gamma)$
(A11)	$I_i(done(\gamma_1;\gamma_2)) \rightarrow [\gamma_1]I_i done(\gamma_2)$
(A12)	$I_i(done(\gamma_1 \cup \gamma_2)) \rightarrow I_i done(\gamma_1) \vee I_i done(\gamma_2)$

Axiom (A0) can be obtained from the definition of $p(\mathcal{A}g)$. If a state such as *proposed(c_i)* holds, the state *proposed* also holds. *Axiom (A1)* makes iterated belief redundant. This is proved in [5] from the KD45 axioms for belief. *Axiom (D-Int)* ensures that an agent's intentions are consistent. *Axioms (Int-A)* and *(Int-B)* are respectively analagous to positive and negative introspection for belief. An agent is aware of its intentions (Int-A) and its non-intentions (Int-B).
Axiom (A2) expresses that if agent i intends A, then it intends to believe A. The reverse implication does not hold (i.e. an agent may intend to believe A but not intend A because it is not capable of bringing about A).
Axiom (A3) defines the simplest relation between intention and belief [3]. Thus, if agent i intends A, then it does not believe A holds. This is weaker than believing $\neg A$, because doing so would result in an inconsistency with axiom (A4).
Axiom (A4) is derived by replacing A with $B_i A$ in axiom (A3) and means an agent does not intend to believe what it already believes.
Axiom (A5) is the contraposition of axiom (A3) and means an agent does not intend A if it believes A already holds.

[1]We may extend the language to associate an agent with roles. For example, c_i*:contractor* could mean c_i acts as a contractor.

Axiom (A6) is obtained by replacing A with $I_i A$ in axiom (A4) and then applying axiom (D-Int). It implies that if i intends something, it does not intend to intend it. Thus, in contrast to beliefs, intentions do not expand nor collapse.
Axiom (A7) is obtained from axiom (Int-A) and allows to simplify alternating beliefs and intentions to only intentions.
We also specify axioms for relating process execution in PDL with beliefs and intentions. *Axioms (A8)* and *(A9)* define that if i intends A (or intends $done(\gamma)$) and it believes that A holds after executing γ, then after γ is performed, i believes A.
Axiom (A10) expresses that if i intends A and it believes that A holds after executing γ, then it will intend $done(\gamma)$.
Axiom (A11) states if i intends a sequence of processes, then it intends the next process in the sequence after each process execution.
Axiom (A12) expresses that i intends either sub-process when intending a complex process consisting of alternative processes. Given the above axioms relating to the modalities in PDL-BI, we can now define axioms regarding message exchange, which is the second component of the first layer in figure 1.

4.4 Communication Axioms

A speech-act consists of an illocutionary force *sa* and a propositional content p [10]. Thus using PDL-BI notation, a CA is a tuple of the form *s.sa(r,A)*, where s is the sender, r the receiver and A a formula representing the propositional content of the act. We assume that sent messages are successfully relayed across the network, implying that the action *s.sa(r,A)* always succeeds. This suggests the following PDL-BI axiom schema:

$$I_s done(s.sa(r, A)) \leftrightarrow [s.sa(r, A)]B_r done(s.sa(r, A))$$

The precondition $I_s done(s.sa(r, A))$ is necessary because a message should not be sent unless the sender intends to send it. It is sufficient for the receiver to presume these intentions in order that the precondition to be itself a primitive plan by which the sender can attain the RE. This causality style of schema is potentially verifiable in a logic that is grounded in machine states, but for our purposes it enables us to separate the FP and RE from the message transport. FIPA-like semantics are re-instated by assuming that the receiver believes that the message transport post-condition entails the message RE, $(B_r done(s.sa(r, A)) \rightarrow B_r RE_s a)$. The FIPA semantics do not assume the sender to intend the RE, but one can take the view that the FP should be strengthened to entail $I_s done(s.sa(r, A))$.

Let *FP(sa)* denote the FP of the CA *sa* being sent and *RE(sa)* denote the RE of *sa*. In general, the FP relates to a sender's intentions to convey some formula and the RE relates to the receiver's beliefs. Thus, we have the following two axioms:

Message Sending: $I_s done(s.sa(r, A)) \rightarrow FP(sa)$.
Message Receipt: $B_r done(s.sa(r, A)) \rightarrow RE(sa)$.

If the sender intends to send a message, then the FP of the message (its CA) should hold, and likewise for the receiver to believe the RE of the message on receiving it. These communication axioms may change according to the properties of the network layer.

4.5 Social Attitudes

The social attitudes, defined at the second layer in figure 1, specify the characteristics of a particular set of agents (e.g.

whether they are sincere, trustworthy or selfish). As a first step, we assume sincere and trustworthy behaviour and in this section we define the axioms that capture such attitudes. These attitudes are implicit in FIPA ACL and in this section we make them explicit, so that they may be relaxed in other scenarios.

Sincerity Axioms: Agents are sincere and the sender does not intend the receiver to believe what it does not believe itself. $I_sB_rB_sA \rightarrow B_sA$ and $I_sB_rI_sA \rightarrow I_sA$. Using axiom (A1) on $I_sB_rB_sA \rightarrow B_sA$, we have $I_sB_rA \rightarrow B_sA$.

Trust Axiom1: If r believes that agent s sent a message to agent r that A holds, then r believes A. Receivers trust the sender. $B_rI_sB_rA \rightarrow B_rA$.

Trust Axiom2: If A is a proposition or a belief formula in *Trust Axiom1*, then r also believes s believes A. $B_rI_sB_rA \rightarrow B_rB_sA$.

Cooperative Axiom: Agents are cooperative and, thus, on receiving a message, they will reply even if it is with a refusal or a rejection. $B_rI_sI_rA \rightarrow (I_rB_sI_rA \vee I_rB_s\neg I_rA)$.

The above axioms can also be related to a message sent or received. For example, the trust axiom can be re-formulated as: $\neg B_r\neg A \rightarrow [s.sa(r,A)]B_rA$. This means that if the receiver r does not believe that A does not hold, then on being informed by s that A holds, r believes A.

Given the above system for an agent's propositional and social attitudes, we can now specify the semantics of CAs in terms of the intentions of a message sender and the receiver's beliefs.

5. SEMANTICS FOR CAS

We analyse the semantics of the most commonly used CAs, as given by FIPA in the SL language, and discuss the incorrectness of these semantics with respect to the intended meaning of the CA. As a remedy, we then provide a corrected, simpler and more intuitive semantics (in table 4). In what follows, we refer to the SL semantics as "FIPA semantics" and we refer to our revised semantics as "BIS" (Belief Intention Semantics). In all cases, s and r respectively denote the sender and receiver, A denotes a formula and γ a process.

Table 4: Semantics for FIPA CAs

CA	FP	RE
$s.inform(r,A)$	B_sA	B_rA
$s.propose(r,\gamma)$	$B_s(B_sI_rdone(s.\gamma) \rightarrow I_sdone(s.\gamma))$	$B_r(B_sI_rdone(s.\gamma) \rightarrow I_sdone(s.\gamma))$
$s.accept(r,\gamma)$	$B_s(B_rI_sdone(r.\gamma) \rightarrow I_rdone(r.\gamma))$, $B_sI_sdone(r.\gamma)$	$B_rI_sdone(r.\gamma)$
$s.reject(r,\gamma)$	$B_s(B_rI_sdone(r.\gamma) \rightarrow I_rdone(r.\gamma))$, $B_s\neg I_sdone(r.\gamma)$	$B_r\neg I_sdone(r.\gamma)$
$s.request(r,\gamma)$	$B_sI_sdone(r.\gamma)$	$B_rI_sdone(r.\gamma)$
$s.agree(r,\gamma)$	$B_sI_rdone(s.\gamma)$, $B_sI_sdone(s.\gamma)$	$B_rI_sdone(s.\gamma)$
$s.refuse(r,\gamma)$	$B_sI_rdone(s.\gamma)$, $B_s\neg I_sdone(s.\gamma)$	$B_r\neg I_sdone(s.\gamma)$
$s.cfp(r,\gamma)$	$B_sI_s($ $done(r.propose(s,\gamma)) \vee$ $done(r.refuse(s,\gamma)))$	$B_rI_s($ $done(r.propose(s,\gamma)) \vee$ $done(r.refuse(s,\gamma)))$

As for FIPA semantics, we give the preconditions (FP) and postconditions (RE) holding, respectively, before sending and after receiving a CA. To be compatible with the way FIPA semantics are expressed, using axiom (Int-A), we prefix the intentions of a sender with its beliefs about those intentions (e.g. $B_sI_sB_rA$ instead of I_sB_rA). For reasons mentioned in section 2, we drop the Gricean conditions from the FP of all the CAs. Although it may not be appropriate to repeat sending an *inform* CA, removing the GC does not affect the meaning of the other CAs. In any case, a sender repeating an *inform* does not change the sender's or receiver's beliefs about the propositional content after the first *inform*. We discuss below the semantics of some of the most salient CAs.

s.inform(r,A). s informs r that A holds. In the FIPA semantics, the FP includes the GC and the fact that the sender believes A. The RE includes that the receiver believes A. As mentioned in section 4.4, B_sA is not strong enough as FP in the FIPA semantics since we need to represent the intention of s to send the message. We could express this as $B_sI_sB_rA$, the sender intends the receiver to believe A. By the sincerity axiom, the FP simplifies to B_sA and by the trust axiom the RE simplifies to B_rA.

FIPA *inform*	FP:$B_sA \wedge \neg B_s(Bif_rA \vee Uif_rA)$ RE: B_rA
BIS *inform*	FP: B_sA RE:B_rA

s.propose(r,γ). s proposes r for s itself to do γ. In the FIPA semantics FP, the sender s will intend to do γ if r intends s to do γ. However, s may not know what r intends and therefore cannot consequently infer that it should intend to do γ. For s to be aware that r intends $done(s.\gamma)$, it must have received an accept to its proposal. As such, FIPA semantics specify that s adopts an intention by being privy to the individual beliefs of r. In our BIS semantics, both the FP and RE specify that s (the proposer) believes that r intends $done(s.\gamma)$, for s to adopt the same intention. Therefore, $B_sI_rdone(s.\gamma)$ is the premise for s to adopt the intention to do γ.

FIPA *propose*	FP: $B_sA \wedge \neg B_s(Bif_rA \vee Uif_rA)$ RE: B_rA where, $A = I_rdone(<s,\gamma>) \rightarrow I_sdone(<s,\gamma>)$
BIS *propose*	FP: $B_s(\mathbf{B_s}I_rdone(s.\gamma) \rightarrow I_sdone(s.\gamma))$ RE: $B_r(\mathbf{B_s}I_rdone(s.\gamma) \rightarrow I_sdone(s.\gamma))$

s.accept(r,γ). s sends an accept proposal to r, for r to do γ. The FIPA semantics for *accept*ing a proposal do not consider the context of sending an accept. As FP, s believes that it intends r to do γ. There is no notion in the FP that s is accepting a proposal that r must have sent. As long as s intends r to do γ, it can send an *accept* without any prior proposal from r or even with a previous refusal from r. Thus, the FIPA FP and RE for *accept* could also hold in other speech-acts such as *tell* and they do not distinguish an *accept-proposal* from them. In our BIS semantics, we include in the FP that the sender believes that r sent a proposal previously and it is up to s to accept it. The other part is the choice of the sender to intend the receiver to do γ[2]. Here our semantics of the *accept* CA show the necessity of having both the FP and RE for specifying the semantics of a CA. The FP and RE involve more than just the sender intending A before sending a CA and the receiver believing A in the RE. For example, for the *accept* CA, only the FP of *accept* includes the belief about a prior proposal.

[2]The same remarks as for *accept* apply to the FIPA semantics for *reject-proposal*. The BIS semantics for *reject* are given in table 4.

FIPA *accept*	FP: $B_s A \wedge \neg B_s(Bif_r A \vee Uif_r A)$ RE: $B_r A$, where $A = I_s done(<r,\gamma>)$.
BIS *accept*	FP: $B_s(B_r I_s done(r.\gamma) \rightarrow I_r done(r.\gamma)),$ $B_s I_s done(r.\gamma)$ RE: $B_r I_s done(r.\gamma)$

s.cfp(r,γ). *s* sends a call for proposal to *r* to do γ. In the FIPA semantics, the FP for a call for proposal includes that both sender and receiver intend the receiver to perform γ if feasible. However, these intentions are premature given that *r* has yet to propose and *s* to accept for *r* to do γ. It does not leave the possibility for refusal or rejection. The rest of the semantics for *cfp* is so complicated that its meaning is unclear. In our semantics, a call for proposal from *s* to *r* is equivalent to a request from *s* to *r* for *r* to make a proposal or a refusal to *s*. Thus *s.cfp(r, γ)* is partly equivalent to *s.request(r, r.propose(s,γ))*. Using our BIS semantics for *s.request(r, γ)*, as shown in table 4, we can specify a call for proposal by *s* as having FP $B_s I_s(done(r.propose(s,\gamma)) \vee done(r.refuse(s,\gamma)))$. This means that *s* intends *r* to either send a proposal or refusal to do γ (for example because *r* cannot do γ). In turn, the RE of a call for proposal is that *r* believes *s* intends *r* to make a proposal or to refuse. The FP and RE of *cfp* can be further expressed in beliefs and intentions by replacing its constituent CAs *propose* and *refuse* with their own BIS semantics.

6. INTERACTION PROTOCOLS

Interaction protocols (the fourth layer in figure 1) can be represented diagrammatically in statechart like notation [7] (for the sake of comprehension) and logically in extended PDL (for verification purposes). Specifically, an interaction protocol consists of state transitions and a hierarchy of states. Let the predicate *one_of* be defined to return *true* if only one of the formulae in its given set holds. Therefore, *one_of({A, A_1, A_2, A_3})* holds if only one of the formula in the set {A, A_1, A_2, A_3} holds.

Sub-states. These are represented using single and double implication between states. For example,
$A_0 \leftrightarrow$ *one_of*$(\{A_1, A_2\})$ expresses that A_1 and A_2 are sub-states of A_0. The double implication ensures that the state A_0 cannot hold unless either A_1 or A_2 holds.

State transitions. The formula $A_1 \leftrightarrow [\gamma]A_2$ expresses a state transition from source state A_1 to target state A_2 through process γ. Double implication in the formula allows to infer the source state after a transition has occurred. To apply our framework, the remainder of this section is concerned with the CNP as a case study of an IP.

6.1 Representation of the CNP

We can express the CNP using CAs and the modalities in PDL. We treat the exchange of CAs as the execution of processes. Our extensions to PDL allow us to express agents executing actions and thus sending CAs. A fragment of the representation of the CNP is given in figure 2. The corresponding formal semantics of the protocol are described in section 6.2. Let *M* denote the manager and, *i* and *j* contractors.

Figure 2 provides the logical theory of most of the CNP in extended PDL. Axioms (1)-(3) specify relations between the states. For example, a *contract_net* state is either *open* or *closed*, where the sub-states of *closed* are given in axiom (2) and those of *open* in axiom (3). A manager may initiate a contract net into a *cfped* state by issuing a call for proposals

contract_net $\leftrightarrow$ *one-of ({ open, closed})* (1)
closed$\leftrightarrow$*one-of({(failed*$\wedge$*informed),cancelled,refused})* (2)
open $\leftrightarrow$ *one-of({deliberating, executing})* (3)
$\neg$ *contract_net* $\leftrightarrow$*[M.cfp(G,*γ_M*)] cfped(M,G,*γ_M*)* (4)
cfped(M,G,γ_M*)*$\leftrightarrow$*(*
[i.refuse(M,γ_M*); M.wait]cfped(M,G,* γ_M*)*
$\vee$ *[i.refuse(M,*γ_M*); timeout?] refused(*γ_M*)*
$\vee$ *[i.propose(M,*γ_i*);C* $\setminus$ *{i}; P*$\setminus$ *{*γ_i*}] proposed(C, P))* (5)
proposed(C, P) $\leftrightarrow$
[j.propose(M,γ_j*);C*$\setminus$*(C* $\cup$ *{j}); P*$\setminus$*(P* $\cup$ *{*γ_j*})]proposed(C,P)*
$\vee$ *[M.deliberate(C,Acc,P)]to-be-accepted(C,Acc,P)* (6)
to-be-accepted(C,Acc,P)$\leftrightarrow$
$\forall i \in$ *Acc [M.accept(*i, γ_i*)] accepted(Acc, P)* $\wedge$
$\forall j \in$ *(C-Acc) [M.reject(*j, γ_j*)]rejected(C-Acc, P)* (7)

Figure 2: A fragment of the CNP in extended PDL

to a group of contractors *G*, only if the interaction has not yet started, leading to the *cfped* state (Axiom 4).

Axioms (5)-(7) define the state transitions of the CNP. Axiom (5) expresses that in a *cfped* state, contractors may refuse the manager's *cfp*, and if the manager receives only refusals by the deadline, the process terminates in a *refused* state. Otherwise, some contractors may refuse whilst others (proposers) send a proposal, leading to *proposed(C, P)* where *C* is the set of proposers and *P* the set of proposals. In the set *P*, each proposal is associated with its proposer (i.e. γ_i is associated to proposer *i*). In axiom (6), further proposals from other proposers are added to the sets *C* (via $C \setminus (C \cup \{j\})$) and *P*. The manager deliberates which proposals to accept and add them to the set *Acc*. *Acc* is the set of proposals which the manager will accept. Thus, in axiom (7) the manager sends an *accept* message to all contractors in *Acc*, and rejections to those in *(C-Acc)*.

6.2 Semantics of the CNP

To specify the semantics of an IP, we must first specify the semantics of its state transitions, which, in turn, lead to a corresponding semantics for its states. On combining the semantics of an IP's states and transitions according to the allowable sequences of transitions defined in the representation of the protocol (figure 2), we obtain an interpretation of the protocol and its semantics (layer 4 in figure 1). This interpretation can then be verified for properties such as correctness, termination or completeness (see section 7.1). First, we specify the semantics of the transitions in the CNP. There are two types of transitions in the CNP, CAs (mostly) and the internal actions of the agents. The semantics of the CAs are defined in section 5 and thus we specify the semantics of the internal actions below before we address the states of the CNP.

6.2.1 Internal Actions.

There are two internal actions in the CNP protocol – *M.deliberate(C,Acc,P)* from a *proposed* state and $i.\gamma_i$ from an *accepted* state. We assume internal actions to be successfully executed. The process $i.\gamma_i$ expresses that agent *i* executes γ_i and its semantics are given in terms of the semantics for extended PDL. For process $M.deliberate(C, Acc, P)$, the set *C* contains those agents which sent a proposal, *Acc* is the set of agents whose proposals *M* will accept and *P* is the set of proposals subscripted with their corresponding proposer. A man-

ager internally performs the process $M.deliberate(C, Acc, P)$ to select which proposals to accept. The semantics of *deliberate* are given in table 5. The precondition requires that M believes that the set C contains those agents which sent a proposal. The postcondition specifies that after a *deliberate* action, the set *Acc* contains the contractors whose proposals M will accept and the set *(C-Acc)* those that will be sent rejections.

6.2.2 Semantics of the States of the CNP.

The interaction states of the CNP can be grouped into three categories: *public*, *shared* and *individual*. A public state is believed by all the agents, a shared state is believed by a particular subset of the group, and an individual state is believed by only one agent. In the CNP, a state s_i is equivalent to $done(\gamma_i)$, where action γ_i triggers state s_i. For example, the state *cfped(M, G, γ)* can be re-written as *done(M.cfp(G,γ))* and likewise for the other states. For the sake of readability, we prefer to name a state as the past tense (e.g. *cfped*) of an action leading to it (e.g. *cfp*), instead of a parameterised *done* (e.g. *done(cfp)*). Let a group of contractors be denoted by G and the manager by M. Let $E_G A$ be read as everyone in group G believes A.

Public States: The public states in the CNP are *cfped, open* and *closed*. These states are believed by the manager and all the contractors in G. We give here the semantics of *cfped*, and in table 5 the semantics of the other public states are be similarly specified in terms of the beliefs of the group GM (where $GM = G \cup \{M\}$). In the state *cfped(M, G, γ_M)*, everyone in the group GM believes *done(M.cfp(G,γ_M))*. That is, $E_{GM} \forall i \in G(done(M.request(i, i.propose(M, \gamma_i))))$. This entails that everyone believes the FP and RE of a call for proposal: $E_{GM} \forall i \in G(I_M(done(i.propose(M, \gamma_i)) \vee done(i.refuse(M, \gamma_M))))$, which can be expressed in only beliefs and intentions by replacing *propose* and *refuse* with their own BIS semantics.

Shared States: In the CNP, the shared states are believed by the manager and a contractor. For example, only the manager and a contractor i sending a proposal to do γ_i believe and mutually believe that contractor i has sent the manager the proposal to do γ_i. The shared states of the CNP are *proposed, accepted, rejected, cancelled, refused, informed and failed*. Their semantics are given in terms of the beliefs of the manager and a contractor. Here we explain the semantics of the *proposed* states, while the semantics of the other shared states are given in table 5.

In the state *proposed(C,P)*, the beliefs between M and each contractor i in C are derived from the semantics of the *propose* CA. The FP and RE of the *propose* CA are: $B_M(B_i I_M done(i.\gamma_i) \rightarrow I_i done(i.\gamma_i))$ and $B_i(B_i I_M done(i.\gamma_i) \rightarrow I_i done(i.\gamma_i))$ leading to the shared belief between i and M in the state *proposed(C,P)*: $\forall i \in C(E_{\{M,i\}}(B_i I_M done(i.\gamma_i) \rightarrow I_i done(i.\gamma_i)))$.

Individual States: In the CNP, the two individual states resulting from its two internal actions are *to-be-accepted* and *completed*, as the private state of the manager and a contractor respectively. In the state *to-be-accepted(C,Acc, P)*, the manager privately believes that it will send an acceptance to all contractors in *Acc*. This state's semantics are: $\forall i \in C\ (B_M(B_i I_M done(i.\gamma_i) \rightarrow I_i done(i.\gamma_i)))$, M believes that all contractors in C sent it a proposal. $\forall i \in Acc\ (B_M(I_M done(i.\gamma_i) \wedge I_M B_i I_M done(i.\gamma_i)))$. M intends that all agents in *Acc* execute their proposal and M intends to let them know about its acceptance. $\forall i \in (C - Acc) B_M(\neg I_M done(i.\gamma_i) \wedge I_M B_i \neg I_M done(i.\gamma_i))$ likewise holds for the agents that M decided to reject.

7. INTERPRETATION OF THE CNP

We can validate our semantics for the CNP and its CAs by proving useful properties of the protocol. We do this by reasoning about the possible paths in the CNP.

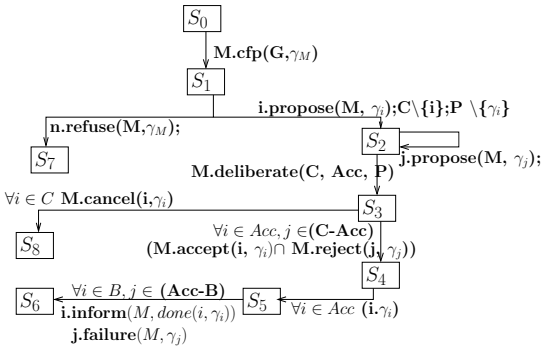

Figure 3: Execution Paths of CNP

Table 5: Semantics of Execution Paths from S_0 to S_6

S_0	$FP_{M.cfp(G,\gamma_M)}$	$\forall i \in G(B_M I_M(done(i.propose(M,\gamma_M)) \vee done(i.refuse(M,\gamma_M))))$
S_1	$RE_{M.cfp(G,\gamma_M)}$	$\forall i \in G(B_i I_M(done(i.propose(M,\gamma_M)) \vee done(i.refuse(M,\gamma_M))))$
	$cfped(M,G,\gamma_M)$	$E_{GM} \forall i \in G(I_M(done(i.propose(M,\gamma_i)) \vee done(i.refuse(M,\gamma_M))))$
	$FP_{i.propose(M,\gamma_i)}$	$B_i \alpha_i$
S_2	$RE_{i.propose(M,\gamma_i)}$	$B_M \alpha_i$
	$proposed(C,P)$	$\forall i \in C(E_{\{M,i\}} \alpha_i)$
	$Pre_{deliberate}$	$\forall i \in C(B_M \alpha_i)$
S_3	$Post_{deliberate}$	$Acc \subseteq C \wedge$ $\forall i \in Acc(B_M I_M B_i I_M done_i)$, $\forall j \in (C - Acc)(B_M I_M B_j \neg I_M done_j)$
	to-be-accepted (C,Acc,γ)	$\forall i \in C(B_M \alpha_i)$, $\forall i \in Acc\ (B_M I_M(done_i \wedge B_i I_M done_i))$, $\forall j \in (C - Acc) B_M \neg I_M done_j \wedge I_M B_j \neg I_M done_j$
	$FP_{M.accept(i,\gamma_i)}$	$B_M \alpha_i, B_M I_M done_i$
	$FP_{M.reject(j,\gamma_j)}$	$B_M \alpha_j, B_M \neg I_M done_j$
S_4	$RE_{M.accept(i,\gamma_i)}$	$B_i I_M done_i$
	$RE_{M.reject(j,\gamma_j)}$	$B_j \neg I_M done_j$
	$accepted(Acc,P)$	$\forall i \in Acc\ E_{\{M,i\}} \alpha_i, (E_{\{M,i\}} I_M done_i)$
	rejected(C-Acc,P)	$\forall j \in$ (C-Acc)$E_{\{M,j\}}(\alpha_j \wedge \neg I_M done_j \wedge \neg I_j done_j)$
	$Pre_{\forall i \in Acc\ (i.\gamma_i)}$	$\forall i \in Acc\ B_i I_i done_i$
S_5	$Post_{\forall i \in Acc\ (i.\gamma_i)}$	$\forall i \in B\ \ B_i done_i$
	completed (Acc, B, P)	$\forall i \in B(B_i done_i \wedge I_i B_M done_i)$ $\forall j \in$(Acc-B)$\wedge B_j(\neg done_j \wedge \neg I_j done_j \wedge I_j B_M \neg done_j \wedge I_j B_M \neg I_j done_j)$
	$FP_{i.inform(M,done_i)}$	$B_i done_i$
	$FP_{j.failure(M,\gamma_j)}$	$B_j(\neg done_j \wedge \neg I_j done_j)$
S_6	$RE_{i.inform(M,done_i)}$	$B_M done_i$
	$RE_{j.failure(M,\gamma_j)}$	$B_M(\neg done_j \wedge \neg I_j done_j)$
	informed(B,$done_i$)	$\forall i \in B(E_{\{M,i\}} done_i)$
	failed(Acc-B, γ)	$\forall j \in (Acc - B)(E_{\{M,j\}}(\neg done_j))$

To this end, figure 3 and table 5 show an interpretation of the CNP from its start (with a call for proposals) to its completion (with *refusals*, *cancellations* or *informs*). Figure 3 includes all possible paths in the execution of the CNP. The path S_0 to S_6 is the longest execution path in the CNP (disregarding iterative proposals), and the semantics for this path are given in table 5. We refer to the states s_i in the CNP as interaction states (e.g. *proposed, accepted, informed*), and to the states S_i in figure 3 as execution states. In table 5, an execution state, S_i, regroups the RE of the action leading to S_i, the semantics of the interaction state s_i and the FP of the next action from S_i. For example, the action *cfp* leads to state S_1 in figure 3, and thus in table 5, the state S_1 groups the RE of *cfp*, the semantics of the *cfped* interaction state, and the FP of the next action *propose*. As before, let M denote the manager, i, j denote contractors and G the group of contractors. In table 5, let $done(i.\gamma_i)$ be abbreviated to $done_i$ and $done(j.\gamma_j)$ be abbreviated to $done_j$. Let α_i denote $B_i I_M done(i.\gamma_i) \rightarrow I_i done(i.\gamma_i)$ and let α_j denote $B_j I_M done(j.\gamma_j) \rightarrow I_j done(j.\gamma_j)$.

7.1 Proving Termination of the CNP

Using table 5 as the semantics of the CNP, we can prove some of its properties (such as termination, completeness, correctness, liveness and consistency). Here, by means of an illustration of our framework, we arbitrarily pick one of these, i.e. termination, and prove that any interpretation of the CNP will terminate.

THEOREM 1. *In a CNP between a manager M and a group of contractors G, the formula* $(cfped(M, G, \gamma_M) \rightarrow [Paths_{ALL}]closed)$ *holds, where the process* $Paths_{ALL}$ *expresses all complete paths of execution in our framework.*

PROOF. We assume $cfped(M, G, \gamma_M)$ and prove $[Paths_{ALL}]closed$. Let the notations in table 5 be used.

We prove that the *closed* state is eventually reached from the *cfped* interaction state. Let the process ρ_i lead to the interaction state s_i and the execution state S_i. The CNP defines what process ρ_{i+1} may follow the process ρ_i. Proving termination for all processes in $Paths_{ALL}$ requires proving that all paths in figure 3 terminate. Thus, we prove that all transitions (actions) in the paths in figure 3 are feasible from their source state. That is, the FP of all processes ρ_{i+1} should hold after action ρ_i and in interaction state s_i. The premise is that the CNP has been started with a call for proposals. The premise holds from the state $cfped(M, G, \gamma_M)$. Then the RE_{cfp} (RE of *cfp*) holds, stating that contractors should either reply with a refusal or a proposal. The cooperative axiom in our framework and the RE of *cfp* allows to trigger the next state. So the process proceeds to the state *proposed* or *refused*. Since *refused* is a sub-state of *closed*, all paths from *refuse* terminate. Using table 5, we now prove that the paths following a *propose* terminate. That is, after execution state S_1, for all actions ρ_i the $FP_{\rho_{i+1}}$ is possible from the interaction state s_i and the RE_{ρ_i}. This can be seen in table 5 where the pre-condition of *deliberate* holds from the $RE_{propose}$.
Also the FP of the action $\forall i \in Acc(M.accept(i, \gamma_i))$ holds since the state *to-be-accepted* includes the belief $\forall i \in C(B_M \alpha_i)$ and $\forall i \in Acc(B_M I_M done_i)$. Similarly, the FP of the *reject* action holds from the beliefs in the *to-be-accepted* state. Thus, both acceptance and rejection processes can occur. Then the precondition of the next action $i.\gamma_i$ can be derived from the beliefs of the state *to-be-accepted*. From table 5 it can then be seen that the FP of both *inform* and *failure* are implied by the beliefs in the state *completed*, leading to the sub-states of *closed*. □

8. CONCLUSIONS

We believe that many of the current problems in enabling agent interactions stem from issues in specifying ACLs and IPs. In particular, there is a lack of consensus on a suitable ACL for agent interaction because of the bewildering array of approaches to formalising an ACL's semantics. Likewise, there is a similarly strong need for formal specification and verification of interaction protocols and their semantics. To highlight these needs and in an attempt to satisfy them, this paper used the contract net protocol as a non-trivial case study. We specified a four-layered framework, separating out the different components for facilitating agent interactions and we proposed the PDL-BI language and its axiomatisation to specify these components. The layers in our framework range from specifying axioms for the propositional and social attitudes of an agent, to specifying the semantics of the ACLs and of IPs. These semantics allow us to verify the properties of IPs such as the CNP and we proved that our interpretation of the CNP terminates.

9. REFERENCES

[1] FIPA. *FIPA Communicative Act Library Specification*. Foundation for Intelligent Physical Agents, http://www.fipa.org, 2002.

[2] R. Goldblatt. *Logics of Time and Computation*. CSLI, 1987.

[3] A. Herzig and D. Longin. A logic of intention with cooperation principles and with assertive speech acts as communication primitives. In *AAMAS*, pages 920 – 927, 2002.

[4] A. J. I. Jones and X. Parent. Conventional signalling acts and conversation. In *Workshop on Agent Communication Languages*, pages 1–17, 2003.

[5] J. Meyer and W. Van Der Hoek. *Epistemic Logic for AI and Comp. Science*. Cambridge Univ. Press, 1995.

[6] J. Odell, H. Parunak, and B. Bauer. Representing Agent Interaction Protocols in UML. In *AOSE Workshop*, pages 121–140, 2001.

[7] S. Paurobally and R. Cunningham. Verification of protocols for negotiation between agents. In *ECAI-15*, pages 43–48, 2002.

[8] J. Pitt and A. Mamdani. Communication protocols in MAS. In *Work. on Specifying and Implementing Conversation Policies*, pages 39–48, 1999.

[9] D. Sadek. A study in the logic of intentions. In *3rd Conf. on Principles of Knowledge Representation and Reasoning*, pages 462–473, 1992.

[10] J. R. Searle. *Speech acts: An essay in the philosophy of language*. Cambridge University Press, 1969.

[11] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, 1981.

[12] M. Wooldridge. Semantic issues in the verification of agent communication languages. *Journal of Autonomous Agents and MAS*, 3(1):9–31, 2000.