Towards Layered Dialogical Agents

Pablo Noriega*, Carles Sierra
Artificial Intelligence Research Institute, IIIA
Spanish Council for Scientific Research, CSIC
08193 Bellaterra, Barcelona, Spain
{pablo, sierra}@iiia.csic.es

Abstract. We present a formalism for the specification of agents within a multi-agent system in which we characterize agents through a layered architecture with bridge rules between formal theories, and multi-agent systems through dialogical frameworks. Concurrent Descriptive Dynamic Logic (an extension of Pceg’s Concurrent Dynamic Logic) is introduced as the specification language to account for the computational interpretation of such multi-agent systems. We illustrate our proposal through the description and implementation of a multi-agent environment: the downward-bidding fish market.

1 Introduction

We conceive agents as computational entities that exhibit complex rational behaviour when acting within a multi-agent system. Such rational behaviour involves several kinds of attitudes: communicational, informational, motivational, etc. We will attempt to integrate such attitudes -which have been thoroughly studied elsewhere, e.g. [23, 11, 10, 4, 18, 1, 24, 20]- by proposing a layered architecture for the construction of agent models, and we will focus on the communicational and social attitudes of agents in order to define multi-agent systems. This layered architecture is based on a crisp separation between attitudes (each one modelled as a theory) and the relations among them (modelled as bridge rules that exchange -or translate- formulas between theories). Interaction among agents will be represented by a special kind of bridge rules between the theories that formalize the social and communicational attitudes of the dialogical agents. Moreover, we take to heart the speech act tenants that communicational exchanges are actions that modify the mental state of agents. And furthermore, that communicational exchanges take place always within a context in which some aspects of ontology need to be shared in order for the agents to react rationally. We introduce the notion of a dialogical framework to formalize the basic ontological and communicational commitments, and we draw upon Dynamic Logic to formalize mental states and their evolution over time as a result of the deliberative and dialogical activity of agents.

Thus, we propose to formalize such conception based on the following explicit assumptions:

(A1) Attitudes can be modelled as theories written in formal declarative languages.

* Visiting researcher from Laboratorio Nacional de Informática Avanzada (LANIA), Rébsamen 80, CP 91000 Xalapa, Veracruz, Mexico.
(A2) Rational behaviour of an agent is the result of explicit interactions between attitudes.

(A3) Bridge rules between theories are adequate to model such interactions.

(A4) Agents are social entities that are to be defined within a multi-agent system.

(A5) Dialogical frameworks are adequate to model the ontological grounding and communication conventions of agents.

(A6) Dynamic Logic is a satisfactory language for the specification of multi-agent systems.

In this paper we present the descriptive elements of a specification language for multi-agent systems based on these assumptions. In fact, we present a framework for the description of multi-agent systems based on a particular kind of layered dialogical agents (Section 3). We then present CDDL, an extension of Concurrent Dynamic Logic, that will be used as a specification language for multi-agent systems modelling. (Section 4). In order to illustrate our proposal we will use a running example throughout the paper: the Spanish fish market, which is introduced in Section 2.

2 The Spanish fish market

The Spanish fish market is a "place" where fish is sold according to the time-honored tradition of downward-bidding auctions [12, 13]. But in a profound sense, the fish market is not merely a "place" but an institution [16] where different agents perform well-established roles under a set of shared conventions (involving strict behavioural protocols, as well as communally understood notions such as price, quality, solvency, or fairness). It is a complex, highly evolved, very effective trading institution and as such it is a good example of a natural multi-agent environment. It constitutes a convenient metaphor to explore structured agent interactions that may even have some practical use, but here we will only describe, and subsequently formalize and present a prototype implementation of the main "scene" of the fish market, its "bidding rounds".

A bidding round presupposes a collection of goods from which a specific item is to be auctioned to a group of potential buyers. The "auctioneer" receives from an admissions officer (the "admitter") the good, its starting price and the list of "buyers" that are going to be involved in that round, she then "opens" the bidding round and calls prices in a descending sequence until a buyer expresses his or her interest to purchase the good. If the potential buyer has a "valid" credit-status (something that the admittance tells the auctioneer) and there is only one standing offer at that price, then the auctioneer "adjudicates" the good and "closes" the bidding round. But if any of these two conditions fail, the auctioneer declares the bid invalid, rises the standing price by 20% and renews the descending price sequence until a new price is accepted by a single able buyer.

With our formalism we will characterize three types of agents: auctioneer, admittance and buyer. We will indicate how their individual attitudes can be formalized and how these attitudes evolve as a consequence of the dialogical interactions between agents. We will also illustrate how these dynamics of theory evolution can be conveniently specified and implemented.

3 Agent Architecture and multi-agent systems

In this Section we shall define a layered agent architecture as a computational entity in which different units, modelling attitudes, are formalized as theories expressed in possibly different languages. Each unit is provided with an initial theory, and a set of inference rules that are used in unit deductions to generate the theory of the unit. These unit-specific languages will contain predicates to represent notions such as Believe, Commit, Know, Declare, etc. Reification of formulas of other such languages, by means of bridge rules\(^2\), will produce instances of those predicates over terms quoting formulas [7, 19] \(^3\).

An agent would then be the theory resulting from a set of unit-theories embedded one into each other by means of bridge rules. The bridge rules incoming to a given unit determine which formulas from other units will extend its theory; likewise, the outgoing connections determine which formulas of the given theory extend other theories. In fact, bridge rules induce a stratified global theory on the set of units, thus conforming the layered agent architecture. Our usage of the term layered is different from its usage elsewhere, e.g. [14]. We want to emphasize that the final agent theory will be stratified, each layer resulting from the action of a particular set of bridge rules.

Note that these languages, units and embeddings are to be tailored for each particular agent in order to express its distinctive features. But it seems desirable, in order to take full advantage of available developments, to keep those theories that formalize a specific attitude as abstract and context-independent as possible while still endowing agents with context-specific knowledge and communicational capacities that allow them to interact successfully with other agents in a specific environment.

As a matter of fact, we will consider only "dialogical" agents, i.e. agents that interact with other agents exclusively through illocutions. Consequently, illocutions may have an effect on the environment—for instance when new entities are admitted into the discourse—but they also have an effect on agent's states of mind—for example, an agent's collection of obligations changes whenever another agent accepts a promise it has made. Thus, each agent will be endowed with a "communicational unit", and we will introduce a special kind of bridge rules that will handle dialogical exchanges among agents. Such communicational bridge rules will be expressed in an illocutory language that will incorporate some basic context-specific elements and may evolve over time.

3.1 Dialogical Frameworks

The shared ontological grounding and illocutory elements that allow agents to interact are made explicit through our "dialogical frameworks" which in turn allow us to shift the attention from the agent architecture to the multi-agent system.

\(^2\) We use the term bridge rule following [7] as a special type of inference rules that have premises in one language and consequences in a possibly different one. See also [2] in this volume.

\(^3\) For instance, the statement that an auctioneer agent \(A\) knows the intention of buyer agent \(B_i\) to buy the good \(\text{cod}=24\) at price \(3500\text{pts}\) could be represented as: \(K(A, \{I(B_i, [\text{buy}(\text{cod}=24, 3500\text{pts})])\})\), where \([\text{buy}(\text{cod}=24, 3500\text{pts})]\) is a term "representing" the formula \(\text{buy}(\text{cod}=24, 3500\text{pts})\), and \([I(B_i, [\text{buy}(\text{cod}=24, 3500\text{pts})])]\) is a term "representing" \(I(B_i, [\text{buy}(\text{cod}=24, 3500\text{pts})])\).
Definition 1. Given a set $\text{Agents}$ of agent names, a set $I$ of illocutionary particles, and a set $\text{Pred}$ of typed predicates, a Dialogical framework is a mapping $DF = \text{Agents} \times \text{Agents} \rightarrow 2^I \times 2^{\text{Pred}}$.

Intuitively, then for each pair of agents $\alpha$ and $\beta$, $DF(\alpha, \beta)$ represents the illocutions and predicates that $\alpha$ can utter to $\beta$ and the illocutions and predicates that $\alpha$ can receive from $\beta$.

Example: The fish market auction requires the next Dialogical framework $DF_{FM}$ for the bidding round scene.

- Participating agents: Auctioneer ($A$), Buyers ($B_i$) and Admissions intermediary (Ad).
- $DF_{FM}(A, B_i) = DF_{FM}(B_i, A) = \{(\text{declare, offer, accept}), (\text{Open, auctionname, Close, auctionname, Sell, good \times price, Buy, good \times price, Sold, good \times price \times buyer, Collision, good \times price, Unsupported Bid, good \times price})\}
- $DF_{FM}(A, Ad) = DF_{FM}(Ad, A) = \{(\text{declare, inquire}), (\text{Open, auctionname, Close, auctionname, Admit, buyer \times auctionname, Credit Status, buyer \times \{valid, invalid\}, Newgood, good})\}
- $DF_{FM}(B_i, Ad) = DF_{FM}(Ad, B_i) = \{(\text{request, concede, deny}), (\text{Admit, buyer \times auctionname, Increase Credit, buyer \times quantity})\}

As a matter of fact, in most dialogical frameworks the extension of predicates needs to be dynamic. This dynamism can be achieved in a rather straightforward manner by introducing specific illocutionary particles such as declare as having effect over time on the extension of the predicates. In our formalism it is captured by the temporal evolution of languages in units.

It is important to note that in addition to the "ground formulas" that can be built with the dialogical framework elements, other more complex formulas may be present in the dialogical exchanges of agents. Formulas that may include operators of different kinds, such as $K, B, Commit, Intend$, etc. are going to be needed to express some of the agent's internal states and may eventually be communicated to other agents.

Thus the layered agent architecture that we propose will expand the dialogical framework according to the specific unit-languages of particular agents. Hence, the following definition:

Definition 2. Given a Dialogical framework $DF : \text{Agents} \times \text{Agents} \rightarrow 2^I \times 2^{\text{Pred}}$, the communication language at time $t$ of agent $\alpha \in \text{Agents}$, noted by $L^\alpha_{C_{\alpha}}$, is upper bounded by the following set of formulas:

$$L^\alpha_{C_{\alpha}} \subseteq \{i(\beta, \gamma, \sigma_t, t) | i \in I, \beta, \gamma \in \text{Agents}, \alpha = \beta \text{ or } \alpha = \gamma, t \in \text{Time}^6, \sigma_t \in \Sigma^\alpha\}$$

and the communication language is then:

$$L^\alpha_{C_{\alpha}} = \bigcup \{L^\beta_{C_{\beta}} | t \in \text{Time} \}$$

where $\Sigma^\alpha = \{[\varphi]|\varphi \text{ is generated by internal bridge rules and unit deductions inside agent } \alpha \text{ from formulas present at time } t - 1 \},$ and $\Sigma^\alpha = \bigcup \{\text{Terms}(DF(\beta, \gamma)) | \beta = \alpha \text{ or } \gamma = \alpha\}$. Terms($DF(\beta, \gamma)$) represents the set of terms naming formulas constructed inside the communication unit of agent $\alpha$ without resource to any formula incoming by means of bridge rules.

3.2 Agent architecture

Our notion of agent involves four constitutive elements:

1. Unit names: Identifiers that denote atomic attitudes.
2. Languages: Declarative and formal with a deductive component.
3. Theories: Sets of language, collection of formulas written in that language and a set of inference rules. Theories are attached to unit names.

Formally, an agent will be given by:

Definition 3. An Agent Structure is a 4-tuple $A = (U, L, T, B)$, where:

1. $U = \{u_k\} \in K$ is a set of unit identifiers.
2. $L = (\mathcal{L}, \Delta)$, is a pair containing a set of finite logical languages $\mathcal{L} = \{L_j\} j \in J$, and a set of inference rules between pairs of languages $\Delta = \bigcup\{\Delta_{j,j'} | j, j' \in J\}$, where $\Delta_{j,j'} \subseteq 2^{L_j} \times L_{j'}$. In particular, when $j = j'$, $\Delta_{j,j'}$ denotes a set of inference rules of the corresponding language; otherwise it denotes a set of bridge rules between two different languages.
3. $T = (\mathcal{M}_L, \mathcal{M}_A, \mathcal{M}_S, \mathcal{M}_B)$ where

(a) $\mathcal{M}_L$ assigns a language to each unit identifier, i.e. $\mathcal{M}_L : U \rightarrow \mathcal{L}$.
(b) $\mathcal{M}_A$ assigns a set of inference rules to each unit identifier, i.e. $\mathcal{M}_A : U \rightarrow 2^4$ such that if $\mathcal{M}_L(u) = L_j$, for some $j \in J$, then $\Delta_{j,j} \subseteq L_j$.
(c) $\mathcal{M}_S$ assigns a concrete signature $\mathcal{M}_S(u) = (\text{Oper, Sort, Func})$ to the language $\mathcal{M}_L(u)$ of each unit identifier $u$, such that $\text{Func} : \text{Oper} \rightarrow \text{Sort}$ gives a type in $\text{Sort}$ to each element in the alphabet $\text{Oper}$.
(d) $\mathcal{M}_B$ assigns a set of formulas (initial unit theory) built upon $\mathcal{M}_S$ to each unit identifier, i.e. $\mathcal{M}_B : U \rightarrow 2^{\mathcal{L}_j}$ such that if $\mathcal{M}_L(u) = L_k$ then $\mathcal{M}_B(u) \subseteq L_k$.

4. $B$ is a mapping that assigns a (possibly empty) set of directed bridge rules to pairs of different units, i.e. $B : U \times U \rightarrow 2^4$, such that:

(i) if $u_1 \neq u_2$, $\mathcal{M}_L(u_1) = L_j$ and $\mathcal{M}_L(u_2) = L_j$ then $B(u_1, u_2) \subseteq \Delta_{j,j}$.
(ii) $B(u, u) = \emptyset$, for any $u \in U$.

---

6 For example, the availability of a new good, e.g. declare($Ad, A, [\text{Newgood}(\text{cod} \neq 23)], 16 : 35$) introduces new terms, e.g. $\text{cod} \neq 23$, and will permit new dialogical exchanges between auctioneer and buyers later on, e.g. declare($Ad, B_i, [\text{Sell}(\text{cod} \neq 23, 1315\text{p}ts)], 17 : 05$).

7 For our purposes, it is enough to consider the set $\text{Time}$ as a linear structure.

---

We consider here ideal reasoning agents that accomplish all their intended conclusions in one instant of time.

Notice that in this definition, even in the case where $u_1 \neq u_2$, $B(u_1, u_2)$ can be empty, denoting that unit $u_1$ has no (directed) link with the unit $u_2$. In this way, a unit $u_1$ is connected to a unit $u_2$ whenever $B(u_1, u_2) \neq \emptyset$. 

---
We will call \( \mathcal{A} \) the class of all possible agents satisfying the structure above. We will call an agent structure **Communicational** with respect to a given dialogical framework, 
\[ DF : \text{Agents} \times \text{Agents} \rightarrow 2^I \times 2^{Pred}, \] 
when it has a (unique) distinguished unit \( C \in U \) for communication purposes, with its corresponding communication language 
\[ M_L(C) \in \{ L^\alpha, \alpha \in \text{Agents} \}. \]

### 3.3 Multi-agent systems

Usually a multi-agent system is described as a computational system consisting of a collection of agents interacting concurrently within a context. We will additionally require that they interact only through the exchange of formulas that respect a given dialogical framework. Hence, we get the next definition:

**Definition 4.** A **multi-agent system** is a 3-tuple \( S = (DF, F, C) \), where

1. \( DF : \text{Agents} \times \text{Agents} \rightarrow 2^I \times 2^{Pred} \) is a dialogical framework.
2. \( F : \text{Agents} \rightarrow \mathcal{A} \) is a mapping from agent identifiers to Communicational agent structures following the agent structure of Definition 3.
3. \( C = \bigcup \{ C_{(\alpha, \beta)} | \alpha, \beta \in \text{Agents} \} \) where \( C_{(\alpha, \beta)} \subseteq 2^{L_{\alpha}} \times L_{\beta} \) is a set of bridge rules between the communication languages of agents \( \alpha \) and \( \beta \).

Note that, in general, the definition of a dialogical framework allows agents in a multi-agent system to have completely independent ontologies and communication languages. The necessary translations between different ontologies and illocutions can be modelled through bridge-rules. When the ontologies and communication languages are shared between a pair of agents, their bridge rules may become the identity translation function.

### 3.4 Example

To have an actual bidding round, three interacting types of agents need to be present, and connected. This collective structure is represented in Figure 1, and definable as*:

\[ S_{FM} = (DF_{FM}, F_{FM}, C_{FM}) \]

\[ F_{FM}(A) = \{(C, E, I), \{LC_A, LE_A, LI_A\}, \Delta_A\}, T_A, B_A\) \]

\[ F_{FM}(AD) = \{(C, E), \{LC_{AD}, LE_{AD}\}, \Delta_{AD}\}, T_{AD}, B_{AD}\) \]

\[ F_{FM}(B_t) = \{(C, E), \{LC_t, LE_t\}, \Delta_{B_t}\}, T_A, B_{B_t}\) \]

In \( S_{FM} \) the only required bridge rule between any two agents \( \alpha \) and \( \beta \) is the identity of common illocutionary formulas:

* Here and in the rest, \( E \) stands for an epistemic unit, \( LE \) for an epistemic language, \( I \) for an intensional unit, \( LI \) for an intensional language. The details of which language, deductive system and initial theory are used, are omitted in this paper.

---

**Fig. 1.** Fish market specification. Arrows mean Bridge rules, balls mean units, squares mean agents, squares and arrows between communication units are concurrent processes.

\[ C_{FM, \alpha, \beta} = \{(l(\alpha, \beta, \sigma, \tau)) \}
\]

Many dialogical exchanges among agents follow a clearly established protocol in which not only are the successive illocutions dependent on the previous ones but also the agent ontologies are to be understood in a well understood way. Furthermore, in our agent model we would like to include abstract deliberative units, languages and theories (proposed, developed or tailored elsewhere), and keep context dependent elements confined to the communication unit and its outgoing and ingoing bridge rules as much as possible. The following example illustrates how dialogical protocols and abstract intentional and epistemic reasoning may be brought together in a neatly layered architecture, in this case the **adjudication process** as performed by the auctioneer agent9.

However, the following are a sample of the kind of bridge rules that the auctioneer agent model includes in order to adjudicate goods only to willing and able buyers. Consider \( \varphi \) to be the formula \( \text{buy}(g, p) \), i.e., the formula that states that a buyer is willing to buy a good \( g \) at price \( p \). Then, the bridge rule schema presented below shows how the auctioneer should interpret a declaration of buyer \( B_t \) to buy at a

9 In this example, in order to make the decision, a temporal persistency modelling inside the epistemic unit is needed. Formulas inside units are not presented, so bridge rules have to be understood as schemas with all variables universally quantified.
given price, as an intention of $B_i$ to buy. The schema says indeed that any declaration of a buyer is considered an intention of the buyer.

$$B_A(C, I) = \{ \ldots, \text{declare}(B_i, A_i, [\varphi], t), I(B_i, [\varphi], t), \ldots \}$$

The auctioneer knows that a buyer is able to buy a good $g$ if the admittance has declared the buyer's credit status "valid", and the good is for sale, i.e. the buyer has previously received an offer to buy that good.

$$B_A(C, E) = \{ \ldots, I(A, t), \text{able}(B_i, [\text{buy}(g, p), [t]), \ldots \}$$

The auctioneer's epistemic theory is aware of the previously declared intentions of other agents.

$$B_A(I, E) = \{ \ldots, I(\alpha, [\varphi], t), K(\alpha, [\text{buy}(g, p), [t]), \ldots \}$$

Finally, the auctioneer adjudicates the good to the buyer that showed the intention to buy and is able to do so. The more complex case of collision-detection\(^\footnote{Two or more buyers declare (simultaneously) their interest in buying at the same price.}\) is not presented here.

$$B_A(E, C) = \{ \ldots, K(A, [\text{buy}(g, p), [t]), \forall B_j \in \text{Buyers}. \text{declare}(A, B_j, [\text{sold}(g, p, B_i)][t]), t') \}$$

The specification language presented in the next Section will provide with the means to formalize the execution dynamics of agents and multi-agent systems.

4 CDDDL. A Multi-agent Systems Specification language

We define Concurrent Descriptive Dynamic Logic (CDDDL) to be used as a specification language for agent modelling. We start with a short reminder on Concurrent Dynamic Logic from which CDDDL is an extension.

4.1 A Reminder of Concurrent Propositional Dynamic Logic

Propositional Dynamic Logic\(^\footnote{Notation: we will use $p, q, \ldots$ to denote atomic propositional variables; $A, B, \ldots$ to denote arbitrary CPDL formulas; $\alpha, \beta, \ldots$ to denote arbitrary programs.}\) is a powerful program logic used as a metalanguage to refer to computer programs. A program can be seen as a dynamic object, that is, an object capable of making the computer pass from one state (the content of the memory registers used by the program) to another. Due to the state change, the truth values of the formulas describing the state also change. The objective of the logic of programs is to create a logical basis to reason about computer programs. $PDL$ provides this by using modal logic as its basis to express changes in truth values due to changes of states. The universe of the Kripke structure is, in $PDL$, a universe of states. Each program has an associated accessibility relation such that a pair of states $(s, t)$ is in that relation if and only if there is a computation of the program transforming the state $s$ into the state $t$. Finally, as in modal logic, each formula is interpreted as a set of states. Note that since we conceive a program as a binary relation between initial and final states, we associate an accessibility relation to every program, thus having a multi-modal language.

An important extension of $PDL$ is Concurrent Propositional Dynamic Logic $CPDL$, where the concurrent executions of different programs is allowed\(^\footnote{An important extension of $PDL$ is Concurrent Propositional Dynamic Logic $CPDL$, where the concurrent executions of different programs is allowed.}\). For a detailed description of $PDL$ and $CPDL$ c.f.\(^\footnote{An important extension of $PDL$ is Concurrent Propositional Dynamic Logic $CPDL$, where the concurrent executions of different programs is allowed.}\) [8, 9].

General Syntax for CPDL Given a set of propositional atomic variables $\Phi_0$ and atomic programs $\Pi_0$, the set $\Phi$ of compound formulas and the set $\Pi$ of compound programs of CPDL are defined as\(^\footnote{An important extension of $PDL$ is Concurrent Propositional Dynamic Logic $CPDL$, where the concurrent executions of different programs is allowed.}\):

1. $T \in \Phi, \bot \in \Phi, \Phi_0 \subseteq \Phi$.
2. if $A, B \in \Phi$ then $\neg A \in \Phi$ and $(A \lor B) \in \Phi$.
3. if $A \in \Phi$ and $\alpha \in \Pi$ then $(\alpha)A, [\alpha]A \in \Phi$.
4. $\Pi_0 \subseteq \Pi$.
5. if $\alpha \in \Pi$ and $\beta \in \Pi$ then $(\alpha \land \beta) \in \Pi, (\alpha \lor \beta) \in \Pi, (\alpha \land \beta) \in \Pi$ and $\alpha^* \in \Pi$.
6. if $A \in \Phi$ then $A^* \in \Pi$.

$(\alpha; \beta), (\alpha \lor \beta), (\alpha \land \beta), \alpha^*$ stand respectively for sequential, undeterministic union, concurrent and iterative computations. Also, $\land, \lor$ and $\leftrightarrow$ are abbreviations with the standard meaning.

General Semantics for CPDL

The semantics of CPDL is defined relative to a structure $M = (S, \{R_k\}, V)$, where $S$ is a set of states, $R_k$ a reachability relation on $S$ for each program $\alpha$, i.e. $R_\alpha \subseteq S \times S^2$, and $V$ an interpretation of formulas, saying in which states they are true, i.e. $V : \Phi \rightarrow 2^S$. A significant difference between $PDL$ and $CPDL$ is that the reachability relation in $CPDL$ is defined on pairs $(s, t)$, where $T \subseteq S$, instead of pairs $(s, t)$ with $s \in T$. This notion captures the intended meaning of the concurrency of operator $\land$, that can lead the computation to one of a set of possible states, each one representing a possible concurrent computation. Hence, the reachability relation for compound programs is defined as:

$$R_\alpha;\beta = R_\alpha \cdot R_\beta$$
$$R_\alpha \cup \beta = R_\alpha \cup R_\beta$$
$$R_\alpha \land \beta = R_\alpha \otimes R_\beta$$
$$R_\alpha^* = R_\alpha^{(*)}$$
$$R_{A^*} = \{ (s, \{s\}) | s \in V(A) \}$$

where

$$s(\{R \cdot Q\}T)$$

iff there exists $U \subseteq S$ with $sRU$, and a collection $\{T_u | u \in U\}$ of subsets of $T$ with $uQT_u$ for all $u \in U$, such that $T = \bigcup_{u \in U} T_u$

$$R \cup Q = \{ (s, T \cup W) | \text{sRT} \text{ and sQW} \}$$
words, to be able to check some properties of multi-agent systems by means of proofs in CDDL, in this paper we just advance the main intuitions and their basic formalisation.

To define CDDL we need to fix the set of atomic formulas and the set of atomic programs. Given an agent \( A = (U, L, T, B) \), with \( L = (L, c, d) \), the set of atomic formulas of CDDL will be defined as the set of "quotation" formulas built upon the languages \( L \) in \( A \) and indexed by the Agent and the unit identifier. More formally,

**Definition 5.** Given a Dialogical Framework \( DF : A \times Agents \rightarrow 2^* \times 2^{pred} \), and an agent structure \( A = (U, L, (M_U, M_A, M_S, M_B), G, B) \) in a multi-agent system \( S = (DF, F, C) \), the set of atomic formulas \( \Phi_{OA} \) for agent structure \( A \) is defined as the following finite set:

\[
\Phi_{OA} = \{ \sigma_{[u]}^A | u \in U, \varphi \in M_L(u) \}
\]

and the set of all atomic formulas in \( S \) is \( \Phi_0 = \bigcup \{ \Phi_{o^A} | o \in Agents \} \).

The sets of formulas \( \Phi_A \) and \( \Phi \) are defined as usual. Given an agent structure \( A \), we next define the set of atomic programs \( \Pi_{OA} \). Atomic CDDL programs will represent deduction steps, inside agents and between agents. From this set of atomic programs the compound program \( \pi_A \) denoting the control of execution of agent structure \( A \) can be defined following the CPD rules for compound program generation.

**Definition 6.** Given an agent structure \( A = (U, L, (M_U, M_A, M_S, M_B), G, B) \), the set \( \Pi_{OA} \) of atomic programs of agent structure \( A \) is defined as the following finite set:

\[\Pi_{OA} = \{ [\Gamma \vdash_k \varphi] \ | \ (\Gamma, \varphi) \in B(u_k, u_i) \} \cup \{ [\Gamma \vdash_{k_l} \varphi] \ | \ (\Gamma, \varphi) \in B(u_k, u_i) \} \}
\]

where \( [\Gamma \vdash_k \varphi] \) is an abbreviation for the quoting function applied to a deduction step.

Having defined the quoting function for formulas, we extend it to sets of formulas and deduction as follows:

**Definition 7.** Let \( \Gamma = \{ \gamma_1, \ldots, \gamma_n \} \) be a set of formulas. Then, \( [\Gamma] = \text{set}([\gamma_1], \ldots, [\gamma_n]) \) and \( [\Gamma \vdash_k \varphi] = \text{proof}([\Gamma], \text{[} \varphi, \text{]} \) \( k_l \) \[\text{[} \varphi \). Where set and proof are names used to construct the term "naming" sets of formulas and proofs.

It is clear then that the access to components of quoted formulas is possible by means of accessor functions. For example, \( \text{conseq}([\Gamma], \text{[} \varphi, \text{]} \) \[\text{[} \varphi \). The execution control for a particular agent structure \( A \) is then defined as a compound program built from this set of atomic programs. We will denote it \( \Pi_A \). Correspondingly, the execution control for a multi-agent system results from the CDDL-composition of agent's execution control programs and the programs associated with the bridge rules for communication among agents:

**Definition 8.** Given a multi-agent system \( S = (DF, F, C) \), the set of execution controls for agents Agents \( \Pi_A = \{ \pi_{o^A} | o \in Agents \} \), and the programs associated to the bridge rules between agents \( \Pi_C = \{ \pi_{o^A} | o \in Agents \} \) \[\text{[} \varphi \). Where \( \pi_{o^A} = \{ [\Gamma \vdash \varphi] \} \) \[\text{[} \varphi \). We define the set of possible execution controls \( \Pi \) for \( S \) as the compound programs that may result from applying the syntactic rules for CPD, defined in Section 4.1, over the set of atomic programs \( \Pi_0 = \Pi_A \cup \Pi_C \),
The usual control program of multi-agent systems will consist of the concurrent execution of the programs associated to agents and bridge rules.

The particular semantics and axiomatics of CDDL correspond to the expected behaviour of the particular type of programs (inference rules). They are out of the scope of this descriptive paper.

4.3 Example

We give here examples of possible execution controls for the agents in the fish market formalisation. Actual differences in control specification would indicate alternative views of what amounts to a deliberative cycle within an agent. In the case of the auctioneer for example we require the full deductive closure of its theories before any new illocution is uttered or heard by the auctioneer, while the admittor has a more "reactive" behaviour.

When a program \( \alpha \) is an atomic program denoting a deductive step, or the undeterministic union of such atomic programs we will denote by \( \alpha^* \) the compound program computing the deductive closure of program \( \alpha \) as defined in [19].

\[
\pi_A = \text{while } \text{auction-open? do } \vdash_{\text{E}} \left( \left( R_E \cap R_C \right) \cap \left( \left( R^*_E \cap R^*_C \right) \right) \right) \end{do}
\]

\[
\pi_{Ad} = \text{while } \text{auction-open? do } \vdash_{\text{E}} \left( R^*_E \cap R^*_C \right) \end{do}
\]

where, in the context of each agent, \( R_{i,j} = \bigcup \{ \alpha | \alpha \in M_A(i) \}, R_{i,j} = \bigcup \{ \alpha | \alpha \in B(i,j) \} \). The meaning of while test? do ... end is the standard in dynamic logic. \( \pi_B \) is analogous to \( \pi_{Ad} \), so we omit it here.

Finally, we make use of the expressive power of CDDL to specify the concurrent execution of agents and bridge rules between pairs of agents. Given that in the fish market example we have \( \text{Agents} = \{ A_1, A_2, \ldots, A_n \} \), the global control of the fish market bidding rounds becomes simply:

\[
\pi = \bigcap \{ \{ \pi_{\alpha} | \alpha \in \text{Agents} \} \bigcup \{ \pi_{\alpha, \beta} | \alpha, \beta \in \text{Agents}, \alpha \neq \beta \} \}
\]

5 Implementation of the example

We have implemented a prototype version of the fish market on PVM\(^{12} \) [6]. In this prototype the implementation follows closely the intuitive description of the bidding round given above. The auctioneer and admittor are resident tasks in the main PVM daemon, while buyers can be spawned in any machine in the network, and can be activated or deactivated externally or by programs. Buyers register and update their "credit line"

\(^{12}\) More recently we have developed other versions of the electronic fish market in which different features and techniques have been explored. A significantly more complete and robust version in Java, FM96.5, is described and documented in http://www.iiti.csic.es/Projects/fishmarket

with the admittor and can participate, if they wish, in a bidding round. When the auctioneer opens a bidding round, participating buyers are "locked" (hence, inhibiting buyers to participate in other scenes, such as updating their "credit lines") until the bidding round is over. Several strategies have been implemented to deal with collisions and unsupported bids. Actual illocutions, in the formal model, are represented as messages exchanged among the PVM software agents. Thus, for example, the auctioneer pricing calls are sent to each buyer together with other good-associated information as a PVM message. Other market information corresponding to the illocutions exchanged by participating agents is presented in the corresponding screens of the different agents. Figure 2 gives a flavour of these interactions and message passing effects. This PVM version is documented in [15].

Fig. 2. Fish market prototype snapshot

6 Final remarks

In this paper we provide a general framework for the construction of agents and multi-agent systems. Two quite independent sources have inspired this theoretical framework: on one hand our work in reflective knowledge systems [19] and, on the other hand, our interest in Computational Dialectics. But, in the background, we have held the honest intention to build actual real-world applications of multi-agent technologies. Hence, the keen reader will find deep connections between the intuitions manifest in our layered
model of dialogical agents with [2] as well as many points of contact with [5, 3, 22], but will probably miss references to our more practical concerns. We will try to succinctly put the more theoretical connections in perspective with our current and future work, as for the methodological and applicative references, here we can only offer our fish market Webpage as a modest alternative to absolute mutism.

In [2] a hierarchical\textsuperscript{13} set of theories constitute an agent and bridge rules are defined between theories in the same way as between units in our framework. The main difference between both papers lays in the fact that they do not study the communication between agents nor the semantics of the dynamics of reasoning. We are now exploring a further generalization of these ideas, in order to treat generalized layered structures—be them agents, interacting groups of agents, or groups of interdependent multi-agent systems—in an abstract uniform way.

The relations between our work and [5, 3, 22] are less immediate, but no less significant. These three works, as well as ours, acknowledge the fundamental importance of dialogical aspects of multi-agent interactions, but each in turn addresses complementary issues in this respect.

While Dignum and Van Linder [5] presents a detailed four level agency model and in particular a very rich treatment of the so called social level, our focus is in the abstract “layering” of these levels. But both papers share a similar understanding of the communication/action relationship and this is reflected in similar operational treatment of specific speech acts. Nevertheless, formal interpretations respond to different preoccupations. We are concerned with a computational interpretation of deliberation and illocutions, hence our dynamic logic approach; while [5] advances a remarkable interpretation of meta-actions as model transforming mappings. It should also be noted that although both papers treat sequences of speech acts, none addresses explicitly the underlying fundamental aspects of dialogical roles, protocols and, in general, the overall discourse structure. We are convinced this is a very important task, and one in which fruitful collaboration is very likely.

Breiter and Sadek present in [3] a concrete theory of rational interaction proposing a specific reasoning method to implement it, while we, in this paper, are concerned with more descriptive formal aspects. But even though the foci are quite different, affinities of the two approaches are more than superficial, as evidenced, for instance, in the implementation of reasoning about action in both papers.\textsuperscript{14}

Likewise, D. Traum, in [22], deals with specific speech acts and some general properties of agents and agent communication that can be readily incorporated into our own framework. Moreover, [22] brings into focus the very important aspect of planning in discourse, which is quite relevant for the description and implementation of complex agent interaction protocols.\textsuperscript{15} Agent interaction protocols is one aspect of multi-agent systems that we have found to be particularly significant, both from a theoretical and an applicational perspective. Our experience with the fish market bidding protocol suggests that intended formal properties, as well as those inherent to a given implementation, result elusive even when institutions are static. In order to deal effectively with complex negotiation protocols and with emergent and ex-post agent interaction protocolization, the kind of tools developed by Traum may prove valuable.

7 Acknowledgements

We are gratefully indebted to ATAL reviewers for detailed and stimulating comments, to Maurizio Giordano for the help in the implementation, and to Juan Antonio Rodríguez and Francisco Martín for their collaboration. We would also like to acknowledge the illuminating explanations of Xavier Márquez, manager of the Blanes Fishermen cooperative market. This research has been supported by the European TMR number PL93-0186 VIM, the Spanish CICyT project SMASH, TIC96-1038-C04-01, and the CONACYT grant [69068-7245]. The final version of the paper has been accomplished while Carles Sierra was on sabbatical leave at Queen Mary and Westfield College, University of London, thanks to the Spanish Ministry of Education grant PR95-313.

References


\textsuperscript{13} The meaning of hierarchical in their work is the same as the layered term for us.

\textsuperscript{14} In [3] reasoning about action is done by means of events, that can be combined by sequence (\(\circ\)) and undeterministic choice (\(\tau\)) and the implementation of the reasoning method is based on a saturation method that terminates. To do so, the number of derivable formulas from a KB must be finite. In the fish market execution control specification we have applied analogous techniques, and, in particular, closure operators require a finite number of possible illocutions, and a finite number of conclusions being generated by inference rules and bridge rules in order to terminate.

\textsuperscript{15} In [21] D. Traum develops a powerful plan execution ontology.
A Rational Agent as the Kernel of a Cooperative Spoken Dialogue System: Implementing a Logical Theory of Interaction

Philippe Bretier  David Sadek

France Télécom
Centre National d'Etude des Télécommunications (CNET—LAA/TSS/RCP)
Technopole Anticipa - 2, avenue Pierre Marzin
F22307 Lannion Cedex France
(bretier, sadek)@lannion.cnet.fr

Abstract. We present the basic components of ARTIMIS, a rational agent based on the implementation of a formal theory of interaction. This theory involves a set of generic axioms which models, in a homogeneous logical framework, principles of rational behaviour, communication, and cooperation. It thus supports the rational unit of an autonomous communicating agent, which, in our case, is the kernel of a cooperative spoken dialogue system. It is expressed in a first-order (multi)modal logic of mental attitudes (belief, uncertainty, and intention) and actions. The implementation consists of an inference engine, which is a theorem prover, specifically designed to reason using the axioms and rules of this kind of logical theory in their syntactic form.

1 Introduction

Designing artificial autonomous agents to be kernels of “intelligent” systems requires the development of formal theories of reasoning and interaction. In this paper, we present the basic components and the implementation of ARTIMIS which is a rational agent based on the implementation of a formal theory of interaction. This theory involves a set of generic axioms which models, in a homogeneous logical framework, principles of rational behaviour, of communication, and of cooperation. It thus supports the rational unit of an autonomous communicating agent, which, in our case, is the kernel of a cooperative spoken dialogue system [25]. It is expressed in a first-order (multi)modal logic of mental attitudes (belief, uncertainty, and intention) and actions.

To implement this kind of theory, two approaches can be adopted: either automate the inference process formalised by the theory, or use the theory as the formal specification of an effective system. We have adopted the first approach since it satisfies, in a more direct way, adequacy, genericity, and maintainability criteria. Obviously, the more complex the targeted behaviours are, the more powerful the theories must be, and therefore the more difficult their mechanisation is. Our implementation consists of an inference engine, which is a theorem prover, specifically designed to achieve reasoning processes using

\[1\] Agent Rationnel à base d'une Théorie formelle de l'Interaction mise en œuvre par un Moïse d'Inference Syntaxique (i.e., Rational Agent based on a Theory of Interaction implemented by a Syntactical Inference Engine).


\[1\] Agent Rationnel à base d'une Théorie formelle de l'Interaction mise en œuvre par un Moïse d'Inference Syntaxique (i.e., Rational Agent based on a Theory of Interaction implemented by a Syntactical Inference Engine).