DIAGAL: A Tool for Analyzing and Modelling Commitment-Based Dialogues between Agents

M. A. Labrie[1], B. Chaib-draa[1], and N. Maudet[2]

[1] Dépt. Informatique et Génie Logiciel, Université Laval
Ste-Foy, PQ, Canada
{chaib,labrie}@iad.ift.ulaval.ca
[2] IRIT, Université Paul Sabatier
Toulouse, France
maudet@irit.fr

Abstract. This paper overviews our currently in progress agent communication language simulator, called DIAGAL, by describing its use in analyzing and modelling automated conversations in offices. Offices are modelled here as systems of communicative action based on dialogue games. Through such games, people in office engage in actions by making promises, stating facts, asking for information, and so on. And through these actions they create, modify, discharge, cancel, release, assign, delegate commitments that bind their current and future behaviors. To make apparent such commitments, we consider here Agent Communication Language (ACL) from the dialectic point of view, where agents "play a game" based on commitments. Such games based on commitments are incorporated in DIAGAL tool, which has been developed having in mind the following questions: (1) What kind of structure has the game? How are rules specified within the game?; (2) What kind of games' compositions are allowed?; (3) How participants in conversation reach agreement on the current game? How are games opened or closed?

1 Introduction

Dialogue games reflects interactions between different participants in dialogue or conversations. In such interactions, each participant intervenes by making utterances, according to a pre-defined set of rules. Typically, the rules define how the dialogue may or must start, what statements may or must be uttered in a given context and, how the dialogue may or must terminate. Such games have found many applications during our history. Thus, they have been used, in ancient and medieval philosophy, for the argumentation and more generally for logical thinking. In modern philosophy, they have been used for the argumentation theory related to the contextual analysis of fallacious reasoning.

Dialogue games have also been applied in computational linguistics, computer science and cognitive science. In computational linguistics, dialogue games have been introduced to explain sequences of human utterances in conversations. Thus, the pioneering work of Levin and Moore [10] introduced the notion

Y. Xiang and B. Chaib-draa (Eds.): AI 2003, LNAI 2671, pp. 353–369, 2003.

of dialogue game as a way of initiating an interaction of a specified type, and of controlling it with a partially ordered set of subgoals. The participants in this game take each her role and attempt to achieve it in the given temporal time. This influential work has found subsequent applications in machine-based natural language processing and generation [9, 11], in human-computer interactions [3, 15]. Recently, dialogue games have been proposed as the basis for "conversation policies" for autonomous software agent communication. To this end, work has focussed on Persuasion dialogues [1]; Negotiation dialogues [2, 17]; agent-team formation dialogues [5]; Commitment dialogues [13]; Dialogues for rational interactions [14], etc. However, none of these approaches has addressed an applicative area with some implementation. This paper attempts to fill this gap by proposing a method for modelling offices as systems of communicative actions based on dialogue games. Through such dialogue games, people in office engage in actions by making promises, stating facts, asking for information, and so on. And through these actions they create, modify, discharge, cancel, release, assign, delegate commitments that bind their current and future behaviors. This paper presents our currently in progress agent communication language DIAGAL (*DIAlogue-Game based Agent Language*) by describing its use in analyzing and modelling automated conversations based on dialogue games.

2 A Dialogue Game Tool Based on Commitments

Commitment-based conversations policies (i.e., general constraints on the sequences of semantically coherent messages leading to a goal) aims at defining semantics of the communicative acts in terms of public notions, e.g. social commitments. In this paper we take a further step in our investigation of this approach by proposing a tool, called DIAGAL (*DIAlogue-Game based Agent Language*) which helps to analyze and model conversations between agents. We have developed this tool having in mind the following questions: (1) What kind of structure has the game? How are rules specified within the game?; (2) What kind of games' compositions are allowed?; (3) How are games grounded? in other words, how participants in conversation reach agreement on the current game? How are games opened or closed? Then, we went further to see how our tool can be used.

2.1 Commitments

As our approach is based on commitments, we start with some details about the notion of commitment. The notion of commitment is a social one, and should not be confused with some psychological notion of commitment. Crucially, commitments are contracted towards a partner or a group. They are expressed as predicates with an arity of 6:

$$C(x, y, \alpha, t, s_x, s_y)$$

meaning that x is committed towards y to α at time t, under the sanctions s_x and s_y. The first sanction specifies conditions under which x reneges its commitment, and the second specifies conditions under which y can withdraw from the considered commitment. In these conditions, the following commitment

$$c_1 = C(Al, Bob, sing(Al, midnight), now, 10, 20)$$

states that agent *Al* is committed towards agent *Bob* to *sing* at *midnight*. If *Al* eventually decides to renege its commitment he will pay the penalty 10. If *Bob* decides to withdraw from this commitment, he will pay 20. We concede that this account of penalties is extremely simple in this version. A more complete account could be similar to the one of Toledo et al. [7]

The notation is inspired from [18], and allows us to compose the actions involved in the commitments: $\alpha_1|\alpha_2$ classically stands for the choice, and $\alpha_1 \Rightarrow \alpha_2$ for the conditional statement that the action α_2 will occur in case of the occurrence of the event α_1. Finally, the operations on the commitments are just creation and cancellation.

$$c_2 = C(Al, Bob, sing(Al, midnight)|dance(Al, midnight), now, 10, 20)$$

and

$$c_3 = C(Al, Bob, music(Bob, midnight) \Rightarrow create(c_2), now, 10, 20)$$

The commitment c_2 captures that the agent *Al* is committed towards *Bob* to *sing* or *dance* at midnight. The commitment c_3 captures that the agent *Al* is committed to contract the preceding commitment (c_2) if agent *Bob* plays *music*. All commitments hold a time concerning when they were contracted (*now*). From now, for the sake of readability, we will ignore the *create* operation. We also permit propositional commitments, that we regard as collections of commitments centering on some proposition p, in the line of [19]. Such commitments are typically the result of assertive moves.

Now we need to describe the mechanism by which the commitments are discussed and created during the dialogue. This mechanism is precisely captured within our game structure. To account for the fact that some commitments are established within the contexts of some games *and only make sense within this context* [11, 14], we make explicit the fact that this commitments are specialized to game g. This will typically be the case of the dialogue rules involved in the games, as we will see below.

2.2 Game Structure

We share with others [4, 8, 14] the view of dialogue games as structures regulating the mechanism under which some commitments are discussed through the dialogue. Unlike [4, 14] however, we adopt a strict commitment-based approach within game structure and express the dialogue rules in terms of commitments. Unlike [8] on the other hand, we consider different ways to combine the structures of the games.

In our approach, games are considered as bilateral structures defined by *entry conditions* (which must be fulfilled at the beginning of the game, possibly by some accommodation mechanism), *success conditions* (defining the goals of the participants when engaged in the game), *failure conditions* (under which the participants consider that the game reached a state of failure), and *dialogue rules*. As previously explained, all these notions, even dialogue rules, are defined in terms of (possibly conditional) commitments. Technically, games are conceived as structures capturing the different commitments created during the dialogue.

To sum up, we have Entry conditions (E), Success conditions of initiator (SI) and partner (SP), Failure conditions of initiator (FI) and partner (FP), and dialogues Rules (R) for each game. Within games, conversational actions are time-stamped as "turns" (t_0 being the first turn of dialogue within this game, t_f the last).

2.3 Grounding the Games

The specific question of how games are grounded through the dialogue is certainly one of the most delicate [12]. Following [16], we assume that the agents can use some meta-acts of dialogue to handle game structure and thus propose to enter in a game, propose to quit the game, and so on. In this case, agents can exchange messages as

$$propose.enter(Al, Bob, g_1)$$

where g_1 describes a well-formed game structure (as detailed above). This message is a proposal of the agent Al to agent Bob to enter the game g_1. This means that games can have different status: they can be *open*, *closed*, or simply *proposed*. How this status is discussed in practice is described in a *contextualization* game which regulates this meta-level communication. As a simple first account of this game, we could adopt the intuitive view of games simply opened through the successful exchange of a propose/accept sequence. However, things are getting more complicate if we want to take account different kinds of combinations. All these kinds of structures are considered within a contextualization game that we do not detail here. Readers interested by these fine aspects can refer to Maudet's thesis [11].

2.4 Composing the Games

As explained before, the possibility to combine the games is a very attractive feature of the approach. The seminal work of [19] and the follow-up formalization of [16] have focused on the classical notions of *embedding* and *sequencing*, but recent works extend this to other combinations [14]. We now detail the games' compositions that we use in our framework. To this end, we precise the conditions under which they can be obtained, and their consequences. Ultimately, such conditions and consequences should be included in the contextualization game we are working on [12].

Sequencing noted $g_1; g_2$, which means that g_2 starts immediately after termination of g_1.
Conditions: game g_1 is *closed.*
Effects: termination of game g_1 involves entering g_2.

Choice noted $g_1|g_2$, which means that participants play either g_1 or g_2 non-deterministically. Not surprisingly, this combination has no specific conditions nor consequences.

Pre-sequencing noted $g_2 \rightsquigarrow g_1$, which means that g_2 is opened while g_1 is proposed.
Conditions: game g_1 is *proposed.*
Effects: successful termination of game g_2 involves entering game g_1.
Such pre-sequencing games can be played to ensure that entry conditions of a forthcoming game are actually established —for instance to make public a conflicted position before entering a persuasion game. Notice that in the case where the first game is not successful, the second game is simply ignored.

Embedding noted $g_1 < g_2$, which means that g_1 is now opened while g_2 was already opened.
Conditions: game g_1 is *open.*
Effects: (conversational) commitments of the embedded games are considered having priority over those of the embedding game.
Much work needs to be done to precisely define this notion within this framework, but this may be captured by constraining the sanctions related to the embedded game to be greater than those of the embedding game ($s_{g2} > s_{g1}$).

Notice that if we want make explicit Initiator and Partner, compositions can be written under the following: $[x, y]g_1; [y, x]g_2$ or $[x, y]g_1|[y, x]g_2$ or $[x, y]g_2 \rightsquigarrow [y, x]g_1$ or $[x, y]g_1 < [y, x]g_2$. In this case $[x, y]g_1$ means that the initiator of g_1 is x and the partner is y.

3 Basic Games

Up to now we have introduced four basic building dialogue games : (1) a "request" game (rg); (2) an "offer" game (og), (3) an "inform" game (ig) and (4) an "ask" game (ag). Sanctions were omitted in our games specifications just for better readability.

3.1 Request Game (rg)

This game captures the idea that the initiator (I) "request" the partner (P) and this latter can "promise" or "reject". The conditions and rules are:

$$\begin{array}{r|l}
E_{rg} & \neg C(y, x, \alpha, t_0) \\
SI_{rg} & C(y, x, \alpha, t_f) \\
SP_{rg} & \text{Nil} \\
FI_{rg} & C(y, x, \neg\alpha, t_f) \\
FP_{rg} & \text{Nil} \\
R_{rg} & C_g(x, y, \mathit{request}(x, y, \alpha), t_0) \\
 & C_g(y, x, \mathit{request}(x, y, \alpha) \Rightarrow \\
 & C_g(y, x, \mathit{promise}(y, x, \alpha) | \mathit{refuse}(y, x, \alpha), t_1), t_0) \\
 & C_g(y, x, \mathit{promise}(y, x, \alpha) \Rightarrow C(y, x, \alpha, t_2), t_0) \\
 & C_g(y, x, \mathit{refuse}(y, x, \alpha) \Rightarrow C(y, x, \neg\alpha, t_2), t_0)
\end{array}$$

Fig. 1. Conditions and rules for the request game

3.2 Offer Game (*og*)

An offer is a promise that is conditional upon the partner's acceptance. To make an offer is to put something forward for another's choice (of acceptance or refusal). To offer then, is to perform a conditional commissive. Precisely, to offer α is to perform a commissive under the condition that the partner accept α. Conditions and rules are in this case:

$$\begin{array}{r|l}
E_{og} & \neg C(x, y, \alpha, t_0) \\
SI_{og} & C(x, y, \alpha, t_f) \\
SP_{og} & \text{Nil} \\
FI_{og} & C(x, y, \neg\alpha, t_f) \\
FP_{og} & \text{Nil} \\
R_{og} & C_g(x, y, \mathit{offer}(x, y, \alpha), t_0) \\
 & C_g(y, x, \mathit{offer}(x, y, \alpha) \Rightarrow \\
 & C_g(y, x, \mathit{accept}(y, x, \alpha) | \mathit{refuse}(y, x, \alpha), t_1), t_0) \\
 & C_g(x, y, \mathit{accept}(y, x, \alpha) \Rightarrow C(x, y, \alpha, t_2), t_0) \\
 & C_g(x, y, \mathit{refuse}(y, x, \alpha) \Rightarrow C(x, y, \neg\alpha, t_2), t_0)
\end{array}$$

Fig. 2. Conditions and Rules for the offer game

3.3 Inform Game (*ig*)

Notice that a partner can be in the disposition of being in accord or agreement with someone without uttering any word. He can also agree by doing a speech act. In this case, he agrees when he can assert a proposition p while presupposing that the initiator has previously put forward p and while expressing his accord or agreement with this initiator as regards p. To disagree is to assert $\neg p$ when the other has previously put forward p. In this game, we assume that the successful termination is when an agreement is reached about the proposition p. The conditions and rules for this couple is the following:

E_{ig}	$C(y, x, p, t_0)$ or $C(y, x, \neg p, t_0)$
SI_{ig}	$C(y, x, p, t_f)$ and $C(x, y, p, t_f)$
SP_{ig}	Nil
FI_{ig}	Nil
FP_{ig}	Nil
R_{ig}	$C_g(x, y, \mathit{assert}(x, y, p), t_0)$ $C_g(y, x, \mathit{assert}(x, y, p) \Rightarrow$ $C_g(y, x, \mathit{assert}(y, x, p) \vert \mathit{assert}(y, x, \neg p), t_1), t_0)$ $C_g(x, y, \mathit{assert}(x, y, p) \Rightarrow C(x, y, p, t_1), t_0)$ $C_g(y, x, \mathit{assert}(y, x, p) \Rightarrow C(y, x, p, t_2), t_0)$

Fig. 3. Conditions and rules for the inform game

3.4 Ask Game (*ag*)

We use "ask" in the sense of asking a question, which consists to request the partner to perform a future speech act that would give the initiator a correct answer to his question. According to these remarks, we propose for the *ask* game the following structure:

E_{ag}	Nil
SI_{ag}	$C(y, x, p, t_f)$ or $C(y, x, \neg p, t_f)$
SP_{ag}	Nil
FI_{ag}	Nil
FP_{ag}	Nil
R_{ag}	$C_g(x, y, \mathit{question}(x, y, p), t_0)$ $C_g(y, x, \mathit{question}(x, y, p) \Rightarrow$ $C_g(y, \mathit{assert}(y, x, p) \vert \mathit{assert}(y, x, \neg p), t_1), t_0)$ $C_g(y, x, \mathit{assert}(y, x, p) \Rightarrow C(y, x, p, t_2), t_0)$

Fig. 4. Conditions and rules for the ask game

3.5 A Simple Use Case Example

To make things more concrete, let us illustrate previous considerations with a request game presented in Fig. 1.

Suppose that agent *Al* and agent *Bob* have entered the request game. *Al* is committed to play a request move towards agent *Bob*, and *Bob* is committed to create a commitment to play a promise or a refuse if *Al* honors his commitment. If *Bob* plays the promise, this will lead to the success condition of *Al* (SI_{rg}) or to failure condition (FI_{rg}) if *Bob* plays a refuse. When the game expires (successfully or not), the commitments that were specialized to this game, those which are dependent on the context "*g*", are automatically cancelled. The others remain as "persistent" commitments.

4 DIAGAL a Dialogue Game Simulator

We have developed a tool (called DIAGAL) which simulates dialogue as a game based on commitments as presented in the previous section while allowing the integration of some future concepts. DIAGAL (*DIAlogue-Game based Agent Language*) aims to be an effective tool of validation as well as a means of analyzing dialogues between agents, diagrams and structures concerning the various games. In this section, we describe the various components of DIAGAL.

4.1 Game Files

As mentioned previously, a game is composed of entry conditions, success conditions, failure conditions and rules of the game. Each of these elements is defined in its own file, adding to the possible information re-use while facilitating the maintainability of the files.

All the files concerning the games are written in XML. That has a major advantage of being easily manageable in liaison with JAVA and the Object-Oriented programming language used to code the simulator. Using XML offers a good way of describing information. The DTD (Document Type Definition), associated with XML files, describes the precise way in which the game designer must create his files. That gives designers and users a mean of knowing if a game is in conformity with its specifications and if it is manageable by the simulator.

The games are loaded when the simulator starts. These games are placed in a list and all the agents can use them to build their dialogues. Note that a game whose files do not answer the criteria specified in DTD will not be loaded.

4.2 Agenda

The agenda is the principal component of DIAGAL. With it, agents and users can follow the effects of the actions on each move on the conversation i.e., check the creation, cancellation, fulfillment, . . . of commitments between the agents. More particularly, an agent's agenda is used by this agent mainly in its process of deliberation on the continuation of the operations to carry out. This structure contains commitments in action as well as propositional commitments deduced from dialogue rules when an action is played. An agenda is in fact a kind of "Commitment Store" where commitments are classified according to time they were contracted.

Each agent has his own agenda which does not contain commitments of all agents which are created in the simulator, but only commitments concerning the agent owner of the agenda being the debtor or creditor. Note that an agenda is private and only its owner has the rights of accessing it.

Because no agent can have access to the agenda of another, whether for writing or reading, we can mention that the agent owner of the agenda is "responsible" for the contents which are found in its own agenda. More particularly, it is the module "Dialogue Manager", intern to the agent, who controls commitments which are added or removed according to various rules of the dialogue games. Concerning commitments in action, a commitment is fulfilled when an action played by an agent corresponds exactly to its description which is in the agenda i.e. all the parameters of this commitment are also present in the action. For example, if an agent is commit to $C(x, y, CloseWindow(x, 9h00))$, action $CloseWindow(x, 9h00)$ does satisfies this commitment, but $CloseWindow(x, 10h00$ does not. In fact, it is the "Dialogue Manager" of the agent which should fulfill or not a commitment when an action is executed.

4.3 Action Board and Game Stack

The action board is mainly a representation of the actions which were played during simulation. It is modelled as a UML sequence diagram. Each workspace has its own board where users can observe the exchanges of messages between agents as well as the time which is attached to these actions. It is represented as an history of the actions carried out relating to each initiated dialogue. In fact, such a board acts as a visual component for the simulator user, to help him understand and analyze what occurred in a dialogue between two agents. Moreover, an agent could use it to remember what actions were played by other agents who communicated with him, and to deliberate thus about the next actions he can play.

The stack is used to keep track of the embedded games during a conversation. Each time a new game is opened, it is placed on the top of the stack inside the related workspace and it becomes the current game of this workspace.

The stack makes it possible to know which game will become active when the top one is closed and withdrawn from the stack. This stack is also used to manage the priority between the games: the top element having more priority over the bottom element.

4.4 Dialogue Workspace

The workspace is an environment which contains all the data which are specific to a dialogue between two agents: games stack, hierarchical relations between agents as well as the actions board. The agenda of an agent is not even found in the workspace since its owner can be implied in several different dialogues and, according to this, in several workspaces.

In Fig. 5, we present an overview of how dialogues work in DIAGAL. As presented in this diagram, we have two communicating agents interacting via the "Dialogue Workspace". They communicate by sending each other some messages (communicative actions) and as such messages are produced, the simulator place them into "Actions Board". In accordance with the current game on the "Game Stack", the "Dialogue Manager" of the agent who sends the message and the agent which received it deduces the appropriate commitments and places them into the appropriate agendas.

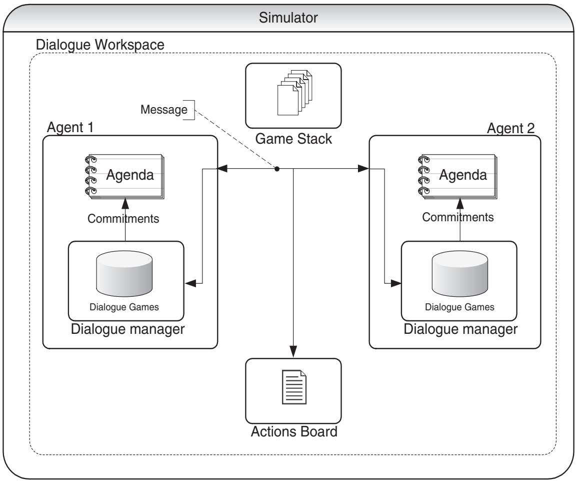

Fig. 5. Simulator overview

In its current form, DIAGAL simulates conversations between a pair of software agents and these conversations are based on dialogue games as presented in the previous sections.

5 Test Case Study: A Summer Festival Organization

We present now an illustrative example which is a first part of an ongoing work on a method for modelling offices as systems of communicative actions based on dialogue games. Through such games, participants engage in actions by making promises, asking for information, stating facts, etc. .... And through these actions, they create, modify, discharge, cancel, release, fulfill, ... commitments that bind their current and future behavior. The illustrative example on which we focus here concerns the organization of a summer festival. This festival which lasts several days consists of a group of artists coming from various areas. We want that all the management task necessary to manage such an event is done between software agents.

5.1 Specific Agents in the Summer Festival Organization

Five various types of agents having some resources to manage were defined for this scenario:

- *AgArtist*: A type of agent representing an artist in the system. An instance is represented by a_i where i is used to indicate that potentially several artists will be present in the simulation. He can accept or refuse invitation regarding the requested fee.

- *AgPlanner*: An instance of $AgPlanner$ (*pl*) is an interface between an agent of type *AgArtist* and the remainder of the system. He is responsible to find the artists to be invited, to manage a budget as well as a schedule. He delegates also some tasks to the agent *AgSecretary*.
- *AgSecretary*: An instance of *AgSecretary* (*sc*) is an interface between an agent of type *AgPlanner* and the resources agents (of type *AgHotels* and *AgTravelAgency*).
- *AgHotels*: An instance of $AgHotels$ (*ht*) is an agent which represents a conglomerate of hotels. He manages a list of rooms, those being able to be reserved by the artists for the duration of the festival.
- *AgTravelAgency*: An instance of $AgTravelAgency$ (*ta*) is an agent which represents a conglomerate of airline companies. He seeks plane tickets for artists.

5.2 A Methodology of Analysis for Commitments Based Task

The method is based on "Partial-Order Planning" to describe the management process of tasks which implies commitments. The principal advantage of this methodology lies in the way of describing efficiently and simply the effects of actions or tasks seen in the form of creation or fulfillment of commitments. The creation of such diagram helps the programmer in his phase of analysis and design of agents as is the case of the coordination of plans for agents of BDI type. It would be possible to use effects as useful preconditions in the relevance of plans as it is case for the plans that JACK agents use, facilitating thus the conceptualization of such agents.

There is a remark which is necessary to bring in connection with the method. The dialogue tasks do not always involve the effects definite on the diagram. For example, the task "Ask artist ar_i for FlightReservation" (c.f. Fig. 6) will not commit ar_i on $C(ar_i, WantTicket(ar_i, date))$ if he refused the request for reservation of a plane ticket. Therefore, if the closing of a dialogue task does not involve the effects hoped, all the actions forming the causal chain rising from this task must be removed. Thus in our example, the causal chain rising from the task "Ask artist ar_i for FlightReservation" should be removed. That corresponds to the suppression of task "FlightReservation for artist ar_i" and the actions *PayTicket* and *SendTicket* as well as the suppression of the effects of these actions. However, the action *ComeToFestival* is not removed since it is protected by an active causal link coming from the invitation of the artist to the festival.

5.3 An Overview of the Dialogue

In our summer festival example, an agent of type *AgPlanner* is the leader of the discussion. It carries out invitations to the agents of type *AgArtist* by taking into account the constraints imposed by its budget. Then he asks artists if: (a) they accept the invitation; (b) they want plane tickets; (c) they want hotel rooms. This agent also transmits the artists' preferences to an agent of type

AgSecretary which is responsible to carry out these reservations with an agent of type *AgHotels* and an agent of type *AgTravelAgency*. After that, the *AgSecretary* agent transmits results of the reservations to *AgPlanner* agent which transmits them, as a confirmation, to the *AgArtist* agent concerned.

Now, we define the meaning of the various tasks presented in Fig. 6.

1. Invitation of artist ar_i:
   Here, the agent of type *AgPlanner* wishes to invite an artist for the Summer Festival. This task consists in opening a "request game" introduced by *pl* towards ar_i. The request concerns the action $ComeToFestival(ar_i, date)$. It is possible that the agents enter some negotiation cycle about the requested action. This means that we could find a sequence of different request-offer

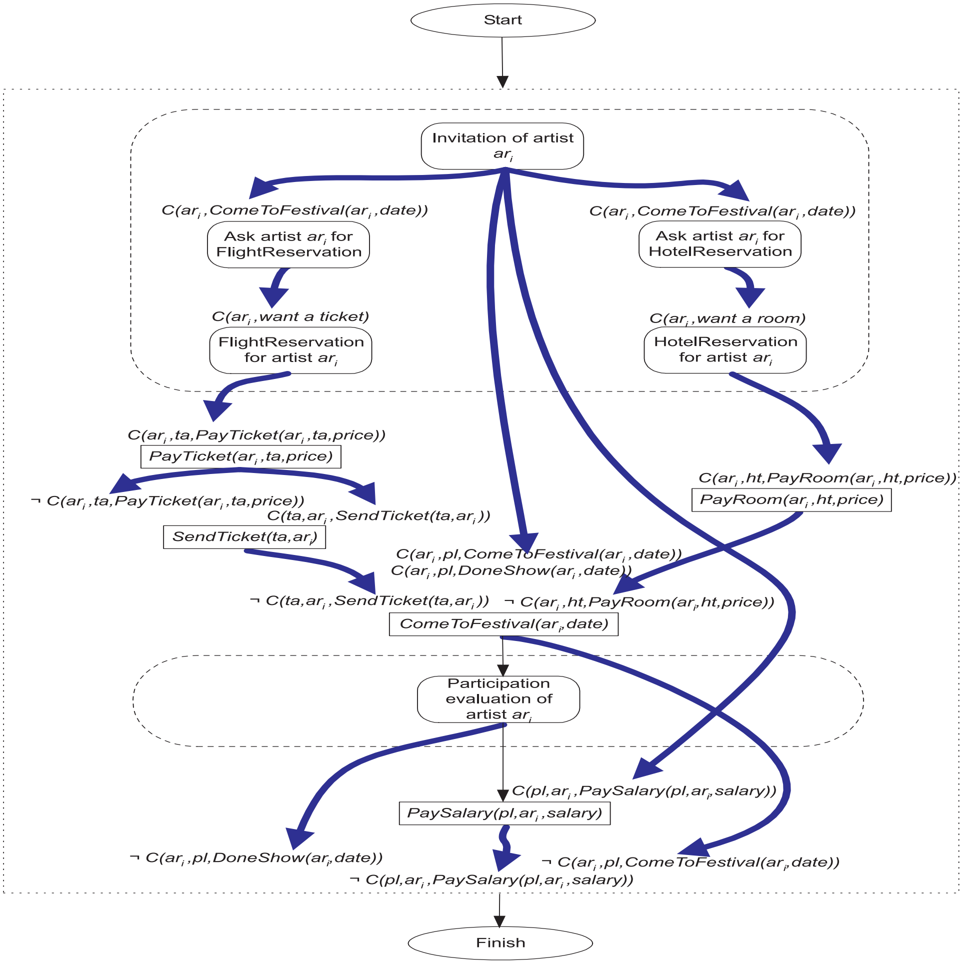
Start
Invitation of artist ar_i
C(ar_i,ComeToFestival(ar_i,date))
Ask artist ar_i for FlightReservation
C(ar_i,ComeToFestival(ar_i,date))
Ask artist ar_i for HotelReservation
C(ar_i,want a ticket)
FlightReservation for artist ar_i
C(ar_i,want a room)
HotelReservation for artist ar_i
C(ar_i,ta,PayTicket(ar_i,ta,price))
PayTicket(ar_i,ta,price)
¬ C(ar_i,ta,PayTicket(ar_i,ta,price))
C(ta,ar_i,SendTicket(ta,ar_i))
SendTicket(ta,ar_i)
C(ar_i,ht,PayRoom(ar_i,ht,price))
PayRoom(ar_i,ht,price)
C(ar_i,pl,ComeToFestival(ar_i,date))
C(ar_i,pl,DoneShow(ar_i,date))
¬ C(ta,ar_i,SendTicket(ta,ar_i))
¬ C(ar_i,ht,PayRoom(ar_i,ht,price))
ComeToFestival(ar_i,date)
Participation evaluation of artist ar_i
C(pl,ar_i,PaySalary(pl,ar_i,salary))
PaySalary(pl,ar_i,salary)
¬ C(ar_i,pl,DoneShow(ar_i,date))
¬ C(ar_i,pl,ComeToFestival(ar_i,date))
¬ C(pl,ar_i,PaySalary(pl,ar_i,salary))
Finish

Fig. 6. Summer Festival Analysis

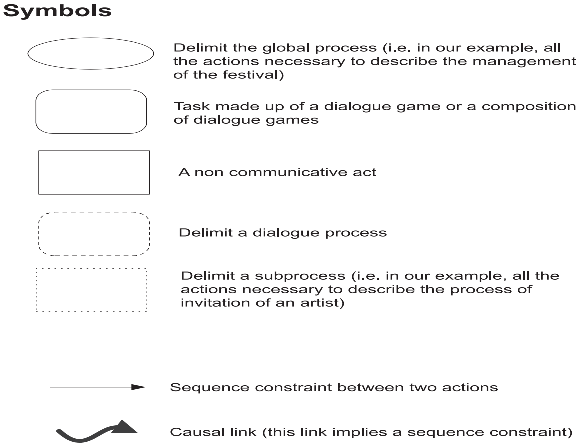
Symbols
Delimit the global process (i.e. in our example, all the actions necessary to describe the management of the festival)
Task made up of a dialogue game or a composition of dialogue games
A non communicative act
Delimit a dialogue process
Delimit a subprocess (i.e. in our example, all the actions necessary to describe the process of invitation of an artist)
Sequence constraint between two actions
Causal link (this link implies a sequence constraint)

Fig. 7. Legend for the Summer Festival Analysis

made by the agents about the requested date. This first phase pre-sequences an offer-request phase were pl offers $PayFee(pl, ar_i, fee)$ and ar_i can counter by requesting an other fee. We use the pre-sequence because the second phase will be ignored if the first one is not successful. Note, that we use the shortcut $(*)$ to stipulate that a sequence can be repeated a number of times, with different games parameters.
Specification: $([pl, ar_i]rg; [ar_i, pl]og)^* \rightsquigarrow ([pl, ar_i]og; [ar_i, pl]rg)^*$

2. Ask artist ar_i for FlightReservation:
The goal of this task is to know if an agent of type $AgArtist$ wants to have a flight reservation. This task simply consists in the opening of an "ask game" by pl towards ar_i or the opening of an "inform" game by ar_i towards pl. These two games concern the proposition $WantTicket(ar_i, date)$.
Specification: $[pl, ar_i]ag|[ar_i, pl]ig$
3. FlightReservation for artist ar_i:
If agent ar_i wants a plane ticket reservation, then this requires a reservation. A "request game" is then addressed by pl to sc about the action $ReserveFlight(sc, ar_i, date)$. Thereafter, sc opens a "request game" with ta concerning the action $ReserveFlight(ta, ar_i, date)$. After this, a request game is addressed by sc to ar_i about the action $PayTicket(ar_i, ta, price)$.
Specification: $[pl, sc]rg$ and $[sc, ta]rg$ and $[sc, ar_i]rg$
4. Ask artist ar_i for HotelReservation:
The goal of this task is to know if an agent of type $AgArtist$ wants to have a room reservation. This task consists in the initiation of an "ask game" by pl towards ar_i or the initiation of an "inform game" by ar_i towards pl about the proposition $WantRoom(ar_i, date)$.
Specification: $[pl, ar_i]ag|[ar_i, pl]ig$
5. HotelReservation for artist ar_i:
If agent ar_i wants a room reservation, this requires a reservation.

A "request game" is then addressed by pl to sc about the action $ReserveRoom(sc, ar_i, date)$. Thereafter, the secretary sc opens a "request game" with ht concerning the action $ReserveRoom(ht, ar_i, date)$. After that, a request game is addressed by sc to ar_i about the action $PayRoom(ar_i, ht, price)$.
Specification: $[pl, sc]rg$ and $[sc, ht]rg$ and $[sc, ar_i]rg$

6. Participation evaluation of artist ar_i:
To complete the global process of invitation, it is necessary for pl to do an evaluation of the action $DoneShow(ar_i, date)$. To do this, pl can initiate an "ask game" with ar_i about the proposition $DoneShow(ar_i, date)$. In an other way, ar_i can open an "inform game" towards the $AgPlanner$ pl also about this same proposition. Thus, one of this two games will trigger the evaluation process.
Specification: $[pl, ar_i]ag|[ar_i, pl]ig$

Note that our model analysis offers some features as represented in Fig. 7. These features are as follows: light arrows show general time constraints between actions. Bold arrows show causal links and implies more specific constraints over actions, specified by commitments. This last type of arrows present the effects of an action or a task on the dialogue at its extremity. These effects are preconditions on the action or task which follow them.

The modelling method, used in this paper, facilitates not only coordination at the level of actions but also coordination at the level of commitments. Indeed, with such method, we can easily follow the evolution of commitments as actions are played.

Each precondition of each step is satisfied by another step. If it is not the case, the causal chain derived from the first one is removed. Every linearization is a possible solution carrying out the goals. Generally, goals in our framework are presented as commitments in action which were fulfilled by the some actions expected by the initiator of the global process, and propositional commitments which hold at the end of the process.

As mentioned in Fig. 7, "dialogue tasks" imply a dialogue game or possibly a composition of some. In our model, DIAGAL, a dialogue game is an interaction mechanism which occurs between two participants. We can see in Fig. 8 between the different pairs of agents which intervenes in our festival example and whose the respective dialogues has been tested using DIAGAL.

6 Conclusion and Further Direction

We have sketched our commitment-based approach for the agent communication language by explaining (1)what is a game in our approach and how this game is structured, what are the rules specified within the game; (2) the kind of games' compositions which are allowed; (3) the ways that participants in conversation reach agreement on the current game and how are games opened or closed. Then we have presented in details our DIAGAL simulator through

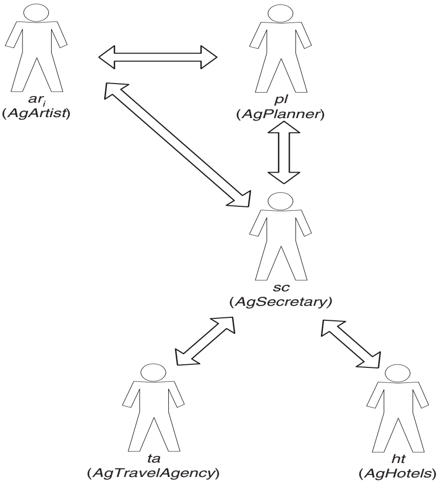

Fig. 8. Communication flow between the different pairs of agents in the Festival Example

the example of summer festival where participants should manage their commitments. We have presented to this end, an approach based on "Partial-Order Planning" which allows designers to describe the management process of tasks implying commitments. We have explained that the principal advantage of this methodology lies in the way of describing efficiently and simply the effects of actions or tasks seen in the form of creation or fulfillment of commitments.

In the future, the simulator will be an indispensable tool allowing at the same moment to simulate conversations among software agents as well as to evaluate metrics on conversations. Among the metrics that we want to address:

1. **Task metrics:**
   - Task completion : i.e., the success rate of a task
   - Task complexity : The minimal number of required interactions for task completion
2. **Commitment metrics:**
   - Commitments release ratio
   - Commitments withdrawn ratio
   - Commitments renege ratio
3. **Qualitative measures:**
   - Agent response delay
   - Dialogue completion delay
   - Utterance accuracy
   - Inappropriate utterance ratio

Acknowledgment

This research is partially supported by Social Sciences and Humanities Research Council of Canada (SSHRC/CRSH).

References

[1] L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In *Proceedings of the 4th Conference on Multi-Agent Systems (ICMAS)*, Boston, 2000. 354

[2] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, Berlin, 2000. 354

[3] T. Bench-Capon, P. E. S. Dunne, and P. H. Leng. Interacting with knowledge-based systems through dialogue games. In *11th International Conference on Expert Systems and Applications Avignon*, pages 123–140, 1991. 354

[4] M. Dastani, J. Hulstijn, and L. V. der Torre. Negotiation protocols and dialogue games. In *Proceedings of the BNAIC*, 2000. 355

[5] F. Dignum, B. Dunin-Keplicz, and R. Vebrugge. Agent theory for team formation by dialogue. In C. Castelfranchi and Y. Lespérance, editors, *Intelligent Agent VII: Proceedings of the Seventh International Workshop on Agent Theories, Architectures and Languages (ATAL 2000)*, pages 150–166, LNAI, 1986, Berlin, Germany, Springer, 2000. 354

[6] F. Dignum and M. Greaves, editors. *Issues in agent communication*, volume 1916 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000. 369

[7] C. Excelente-Toledo, R. A. Bourne, and N. R. Jennings. Reasoning about commitments and penalties for coordination between autonomous agents. In *Proceedings of Autonomous Agents*, 2001. 355

[8] R. F. Flores and R. C. Kremer. A formal theory for agent conversations for actions. *Computational intelligence*, 18(2), 2002. 355

[9] J. Hulstijn. *Dialogue models for inquiry and transaction*. PhD thesis, University of Twente, The Netherlands, 2000. 354

[10] J. Levin and J. Moore. Dialogue-games: meta-communication structure for natural language interaction. *Cognitive science*, 1(4):395–420, 1978. 353

[11] N. Maudet. *Modéliser les conventions des interactions langagières: la contribution des jeux de dialogue*. PhD thesis, Université Paul Sabatier, Toulouse, 2001. 354, 355, 356

[12] N. Maudet. Negociating games —a research note. *Journal of autonoumous agents and multi-agent systems*, 2002. (submitted). 356

[13] N. Maudet and B. Chaib-draa. Commitment-based and dialogue-game based protocols–new trends in agent communication language. *The Knowledge Engineering Review*, 17(2):157–179, 2002. 354

[14] P. McBurney, S. Parsons, and M. Wooldridge. Desiderata for agent argumentation protocols. In *Procceedings of the First International Conference on Autonomous Agents and Multi-Agents*, 2002. 354, 355, 356

[15] D. Moore. *Dialogue game theory for intelligent tutoring systems*. PhD thesis, Leeds Metropolitan University, England, 1993. 354

[16] C. Reed. Dialogue frames in agent communication. In *Proceedings of the Third International Conference on MultiAgent Systems (ICMAS)*, 1998. 356

[17] F. Sadri, F. Toni, and P. Torroni. Logic agents, dialogues and negotiation: an abductive approach. In M. Schroeder and K. S. A. 2001), editors, *Symposium on Information Agents for E-Commerce, AI and the Simulation of Behaviour Conference*, York, UK, 2001. AISB. 354

[18] M. P. Singh. A social semantics for agent communication language. In *[6]*, pages 31–45. 2000. 355

[19] D. Walton and E. Krabbe. *Commitment in dialogue*. State University of New York Press, 1995. 355, 356