A logic-based approach to modeling interaction among computees (preliminary report)

Paolo Torroni[1], Paola Mello[1], Nicolas Maudet[2], Marco Alberti[3], Anna Ciampolini[1], Evelina Lamma[3], Fariba Sadri[2], and Francesca Toni[2]

[1] DEIS, Università di Bologna
Viale Risorgimento 2, 40136 Bologna, Italy
{ptorroni,pmello,aciampolini}@deis.unibo.it
[2] Department of Computing, Imperial College
180 Queens Gate, SW7 London, UK
{maudet, fs, ft}@doc.ic.ac.uk
[3] Dipartimento di Ingegneria
V. Saragat 1, 44100 Ferrara, Italy
{malberti,elamma}@ing.unife.it

Abstract. *Computees* are computational entities interacting in the context of global and open computing environments. The focus of this preliminary report is on the interactions among computees that form a society, and on the definition of a computational logic-based architecture for computee interactions. We propose a layered architecture where the society defines the allowed interaction protocols, which on their turn are defined by means of constraints. The semantics of communicative acts is given in terms of commitments. In order to explain the ideas, we adopt as a running example a resource exchange scenario.

1 Introduction

Computees are computational entities interacting in the context of global and open computing environments [SOC]. They are abstractions of the entities that populate global computing environments. Computees have their own knowledge, capabilities, resources, objectives and rules of behaviour. Each computee typically has only a partial, incomplete and possibly inaccurate view of the society and of the environment and of the other computees, and it might have inadequate resources or capabilities to achieve its objectives. Computees are characterized by exhibiting reasoning abilities, based on a declarative representation of knowledge and on computational logic-based functionalities (e.g. abduction, learning, planning, and so forth). These entities can form complex organizations, which we call *societies of computees*. Since the idea of computee is in a way a specialization of the idea of agent, but with a stress on its computational activities and reasoning capabilities, in this work we will use interchangeably the terms agent and computee.

The focus of this preliminary report is on the interactions among computees that form a society, and on the definition of a computational logic-based architecture. We propose a layered architecture. At the bottom level we put the

Platform used to implement the system and make computees communicate. We will not say anything about the platform layer.

On the top of it, we have what we call the *Computee Communication Language* (CCL), which defines syntax and semantics of communicative acts, and then the *Protocols* layer which determines the set of allowed sequences of communication exchanges in the society. We model protocols by means of constraints. The *Society* is at the top level and defines the set of allowed interaction protocols. The semantics of communicative acts is given in terms of commitments.

Along with the "horizontally" layered structure of the interaction architecture, we have a "vertical" notion of *Scenario*, that we use to put things into context and explain the ideas by means of a running example. The scenario that we use in this report is inspired by the work done by Sadri et al. [STT02], where the authors propose a multi-agent solution to a resource exchange problem, presenting a framework ($\mathcal{N}$-system) based on abductive logic programming, where agents initiate negotiation dialogues to exchange resources. We can imagine that in this scenario computees are the software counterpart of physical agents, but we do not concretely model the aspects related to their physicalness.

The main idea that differentiates this architecture from other similar ones are: (*i*) the use of computational logics to model and give semantics to computees and to their interactions, and (*ii*) the use of integrity constraints to express protocols. Our aim is at some point to be able to propose an operational model for computees and for their societies. In this report we already sketch some bits of a possible model of interaction for computees. Building on previous work on abductive logic-based agents, we define the computee's knowledge as an abductive logic program, and the computee interactions as the result of a local reasoning based on an abductive proof procedure. As a result, in our view, commitments and communicative acts can both be expressed as abducible predicates, and they can be part of integrity constraints.

In this work, after introducing the scenario, we follow a top-down approach to describe the architecture. We dedicate a section for each of the four layers, starting from the society level.

2 Scenario: agents that exchange resources

In this section we briefly recall the resource exchange scenario defined in [STT02]. Let us consider an agent system where agents (computees) have *goals* to achieve, and in order to achieve them they use plans. Plans are (partially) ordered sequences of actions. In order to execute the actions, computees may need some resources. An action that requires a resource r is said to be *infeasible* if r is not available to the agent that intends to execute it. Similarly, infeasible is also a *plan* that contains an action which is infeasible, and so is the *intention* of an agent, containing such plan[1]. The resources that agents *need* in order to perform an action in a plan but that they do not possess are called *missing* resources.

[1] In [STT02] *plans* are modeled as part of the agent *intentions*.

The *resource reallocation problem* is the problem of reducing a possibly non-empty set of missing resources of an agent to the empty set (*r.r.p.* of an agent), and in an agent systems, it is the problem of solving all the *r.r.p.*s of all the agents (*r.r.p.* of an agent system).

The scenario that we propose is about resource reallocation. We consider a society of computational logic-based agents that have goals, plans to achieve the goals, and that all together must solve a resource reallocation problem in order to make their own plans feasible. The exchanges made to achieve a better reallocation of resources are determined by means of negotiation dialogues. As computees are computational entities (pieces of software), we do not model the physical counterpart that they potentially represent.

In particular, what we mean when we talk about *resource* is in fact only an abstract entity, identified by its *name*, which possibly symbolizes a physical resource such as a nail or a hammer. We do not explicitly model the actual delivery of physical resources either, but we assume that communicative acts may come together with *commitments* that at some point agents must fulfill once the act is made. Based on the idea of commitments, we "institutionalize" the agent interactions, and we associate computees societies with institutions.

3 Societies and Protocols

Agent society modeling is a major issue in the field of multi-agent systems. Some of the different approaches adopted to deal with society modeling are the following:

- Cooperative Problem Solving, where cooperation might be presumed or obliged; this kind of approach can be based on the Contract Net or on other methods;
- market models (negotiation, individual interests, partnership formation, concurrency etc.) mainly tackled using game-theoretic appoaches or based on economics (Wellman, Sandholm, ...)
- a more foundational model based on dependency networks (Castelfranchi, Sichman, Luck, Ossowsky), recently enlarged to a more sociological approach, e.g. work by Panzarasa and Jennings, an organizational model (Malone), based then on roles (Mylopoulos-Li; Demazeau; Chaib-draa)
- a teamwork model, benevolence is presumed but should be dynamically managed (Cohen, Jennings, Sonnenberg, Keplicz, ...) nowadays more detailed with individual and pro-social interests (Grosz Kraus, Hogg, ...)
- a deontic model, based on obligations, authorizations, committments (Castelfranchi, Jones-Sergot, Dignum, Carmo, ...)
- finally, there are the reactive and evolving/auto-organizing models, e.g. Ferber, Alife
- evolving organizations (Muller)

The approach to the social "relation" gets blurred with the consequent "type" of society: open/closed; centralized/decentralized; having common or individual

goals. This is only one of the possible interpretations: different works often get overlapped, therefore different groupings are possible.

In our proposal, we rely on a functional definition of a society, implemented through a management infrastructure that supports the definition of roles, constraints on communicative acts (protocols), operations for joining a society and for exiting the society.

In this project, we define computees as social entities that interact with one another, therefore they must be able to *behave socially*.

The situations where computees interact may involve:

- commitment (promises of future actions, or other forms of obligations among participants)
- delegation (computees may act on the behalf of another computee)
- repetition (the same type of interaction is performed repeatedly)
- risk (risk may arise from incomplete knowledge, non-deterministic interactions among computees and with the environment, unpredictable events, etc.)
- ...

These characteristics require a mechanism establishing and enforcing conventions that standardize interactions. Such mechanism can be seen as an *institution*. The concept of institution in multi-agent systems has recently received increasing attention, as it enables the specification and definition of interactions in a flexible and general framework (see, for example [EdlCS02,NS02]). The basic aim of an institution is to facilitate, oversee and enforce commitments among computees.

We suppose the institution is defined by specifying:

- roles
- rules (allowed actions, communication protocols, social commitments)
- operations to join and exit the society

In this work we will only focus on protocols and commitments, but we believe that also the other components of this specification can be easily modeled following a similar approach.

A role is defined by three attributes: responsibilities, permissions and protocols [WJK99]. Responsibilities determine the functionalities. Permissions identify the resources that are available to that role in order to enforce its responsibilities. Protocols identify the way it can interact with other roles.

In a society each computee plays one or more specific roles. Society rules express general, global (supra-role) requirements. These rules can be spread over all roles and protocols, express constraints between roles or implicit rules that moderate the interactions between members.

Global rules allow to identify whether and when to allow new computees to enter the society, and once accepted what their position should be, and which behaviours should be considered legitimate and must be prevented.

To enable high flexibility and generality, we believe that a suitable way of modeling societies is through the specification of protocols. A protocol is assumed in its most general definition, i.e., the set of rules of the interaction that describe what actions each agent can take at each time.

Current formalisms for modeling protocols (e.g. Petri-Nets or finite state machines), specify protocols as legal sequences of actions. In this way, protocols are over-constrained and this affects autonomy, heterogeneity, opportunities, exceptions [YS02b]. According to Yolum and Singh: "Participants must be constrained in their interactions only to the extent necessary to carry out the given protocol and no more". If we follow this approach, computees will be able (at least potentially) to act as they please, provided that they obey the restrictions of the society they belong to and the protocols they must follow.

In a logic-based multi-agent system, we can specify protocols as sets of constraints on the social behaviour.

When computees join a group, they join one or more roles, thereby acquiring restrictions on how can act and, in particular, communicate. Since protocols are expressed in terms of constraints, computees can be tested for compliance on the basis of their communications. Our approach could rely on a "integrity constraint - based semantics" for specifying protocols. The motivations for adopting this approach are the same supporting commitments and committed-based semantics in [YS02b].

3.1 Societies in the resource exchange scenario

A society in this scenario is a set of agents that can communicate by means of request dialogues, i.e., sequences of *dialogue moves*, following a protocol as it is defined in the underlying layer. In this simplified scenario, we do not assume that agents have roles that differentiate any of them from the others. As anticipated in the previous section, if we consider the society in terms of *institution*, we can consider the dialogue moves "as a kind of institutional actions, that is, actions that are possible on the basis of a set of conventions and regulations, and whose effect is to bring about an institutional effect" [CFV02]. This is why in the sequel will talk about *commitments* in association with dialogue moves.

State (I). The only move that is allowed is a *request* move, after which the next state is R.
State (R). The allowed moves are *accept* and *refuse*; in both cases, after either move the next state is F.
State (F). No further continuation is allowed (the dialogue is terminated)

Fig. 1. State transitions in the *request dialogue protocol*

3.2 Protocols in the resource exchange scenario

The only communication protocol allowed in this sample society is the *request dialogue protocol*, that that we define by a state machine. A request dialogue is initiated by an agent, say x, that needs a resource r, if there exists an agent y to whom x still has not requested r. In Figure 1 we define the request dialogue protocol by listing the transition rules (there are three states, of which I is the initial state, and F is the final state)

We can express this protocol in terms of integrity constraints, as it is shown in Figure 2.[2]

possible moves from R:

$$tell(X,Y,accept(give(R)),D,T) \rightarrow \exists tell(Y,X,request(give(R)),D,T'),T' < T$$
$$tell(X,Y,refuse(give(R)),D,T) \rightarrow \exists tell(Y,X,request(give(R)),D,T'),T' < T$$

F is a final state:

$$tell(X,Y,accept(give(R)),D,T) \rightarrow \nexists tell(Y,X,Move,D,T'),T' > T$$
$$tell(X,Y,accept(give(R)),D,T) \rightarrow \nexists tell(X,Y,Move,D,T'),T' > T$$
$$tell(X,Y,refuse(give(R)),D,T) \rightarrow \nexists tell(Y,X,Move,D,T'),T' > T$$
$$tell(X,Y,refuse(give(R)),D,T) \rightarrow \nexists tell(X,Y,Move,D,T'),T' > T$$

I is an initial state:

$$tell(X,Y,request(give(R)),D,T) \rightarrow \nexists tell(Y,X,Move,D,T'),T' < T$$
$$tell(X,Y,request(give(R)),D,T) \rightarrow \nexists tell(X,Y,Move,D,T'),T' < T$$

Fig. 2. Integrity constraints expressing the *request dialogue protocol*

The whole set of constraints or part of it could be embodied in each agent in the society. In order to check at runtime that a protocol is not violated by any agent interaction, we could use a framework such as the one presented in [CLMT02].

4 Computee Communication Language (CCL)

There are several important issues to be considered, when we get to defining a *Computee Communication Language* (from now on, *CCL*):

- which primitives have to be included into the CCL language?
- which CCL semantics?
- how to implement CCL?

[2] In Figure 2, $Move$ represents any expression in the content language **N** (see below).

Which primitives for the CCL. Existing work on ACLs usually defines them as wrapper languages, in that they implement a knowledge-level communication protocol that is unaware of the choice of content language and ontology specification mechanism. This is needed to allow heterogeneous agents to communicate using the designed ACL. However, different choices are possible when deciding what should be expressed in the wrapper language, and what is delegated by the wrapper language to the content language. For instance, a computee A could tell a computee B to achieve a goal X in a couple of different ways (this example is taken from [Vas98])

- `achieve(A,B, goal(X))`
- `tell (A,B, achieve (goal (X))).`

In the first case, a specific `achieve` performative is used; this way of expressing the communicative act "achieve" could be chosen, for instance, when we don't assume that the content language is rich enough to express the concept of achieving.

In the latter case the achieve performative is pushed into the content language, thus eliminating the need for the same in the ACL.

The question of where to place the dividing line between the wrapper language and the content language is crucial if a social (constraint-based) approach is chosen for the definition of the CCL. According to this approach, computees' interactions should be *verifiable* (see Yolum and Singh [YS02b]), i.e., the social infrastructure should be able to detect if computees are not complying to interaction protocols by only observing interactions. This implies that the nature and meaning of interactions should be understandable by the infrastructure, which could be achieved by assigning the semantics of communicative acts at CCL level. On the other hand, requirements of openness and heterogeneity of computees suggest avoiding too strict specifications for the wrapper level. This question is directly related to the way we model the individual computee.

Semantics of the CCL. Once primitives have been chosen for the CCL, in order to give semantics to each of them, we could translate each CCL primitive into a (single) FIPA ACL performative (e.g., *inform*) and then rely on its given BDI semantics. This approach is viable, but still suffers of the already mentioned drawbacks on openness and heterogeneity.

As in the social approach to ACL semantics, defining the semantics of the Computees Communication Language (CCL) as constraints over communicative acts has the clear advantage in that it allows much more heterogeneity and openness than the mentalistic approach does, since it makes no assumptions on the internal structure of the computees.

This approach can rely on a social infrastructure that handle protocols/constraints: with regard to computees communication, it has to check, for instance, whether or not a constraint on a communicative act is violated. The same infrastructure (with the same mechanisms), could also take into account higher level architectural issues. As an example, in an e-commerce society, if two computees agree

on a purchase and commit to it, and if afterwards the buyer refuses to pay, the initial commitment is violated: in this case, the infrastructure, following the society rules, could undertake some particular action, as a reaction to this event (e.g., the buyer is sent off the society).

If the Computees Communication Language is based on Logic Programming and constraints over abducibles, its declarative semantics can be given in terms of logical entailment. Therefore, the operational support of the underlying infrastructure must provide the appropriate proof procedure to ensure the compliance with established protocols.

5 Computee Communication Language (CCL) in the resource exchange scenario

In our running example, we define the CCL by defining two languages:

- the *communication* language, and
- the *content* language, which we call **N**, standing for Negotiation (content) language.

The communication language defines dialogue moves μ as ground terms, having the following format:

$$\mu = tell(Sender, Receiver, \epsilon, Dialogue, Transaction\_time)$$

where *Sender* and *Receiver* are atoms identifying agents, ϵ (the subject) is an expression in the content language, *Dialogue* is an atom which represents a unique dialogue identifier, and $Transaction\_time$ is a positive integer. In [STT02] the authors propose an operational model for negotiating agents ($\mathcal{N}^+$-agents), based on abductive logic programming, where the communication acts or dialogue moves are modelled as abducible predicates. In the $\mathcal{N}^+$-agent operational model, the last two parameters of such predicates, *Dialogue* and $Transaction\_time$, are assigned automatically by the agent cycle, and the agent has not direct control over them.

The content language defines expressions ϵ as:

$$\epsilon \in \{request(give(R)), accept(give(R)), refuse(give(R))\}$$

where R is an atom identifying a resource.

A dialogue move μ could be *syntactically* mapped into a communicative action in a traditional ACL such as FIPA ACL [FIP01]. We give an example for a dialogue move

$$\mu = tell(Sender, Receiver, request(give(R)), Dialogue, Transaction\_time)$$

```
(request
  :sender (agent-identifier :name Sender)
  :receiver (set (agent-identifier :name Receiver))
```

```
:content
       request(give(R))
:language N)
```

Time and dialogue identifier are not mapped here because they are not present in the FIPA syntax, and because in [STT02] we assume that they are generated by the agent cycle.[3]

The semantic of this move is defined in FIPA in terms of mental states:

$$
\begin{array}{l}
< \texttt{Sender}, \texttt{request}(\texttt{Receiver}, \texttt{give}(\texttt{R})) > \\
\quad \texttt{FP} : \texttt{FP}(\texttt{give}(\texttt{R}))[\texttt{Sender} \setminus \texttt{Receiver}] \\
\qquad \wedge\ \texttt{B}_{\texttt{Sender}}\ \texttt{Agent}(\texttt{Receiver}, \texttt{give}(\texttt{R})) \\
\qquad \wedge\ \neg \texttt{B}_{\texttt{Sender}}\ \texttt{I}_{\texttt{Receiver}}\ \texttt{Done}(\texttt{give}(\texttt{R})) \\
\quad \texttt{RE} : \texttt{Done}(\texttt{give}(\texttt{R}))
\end{array}
$$

where

- FP denotes the *feasibility preconditions* or the act, i.e., the conditions that have to be satisfied for the act to be planned;
- RE denotes the *rational effect* of the *request* move, i.e., the reasons for which it is selected;
- $\texttt{FP}(\texttt{give}(\texttt{R}))[\texttt{Sender} \setminus \texttt{Receiver}]$ denotes the part of the FPs of give(R) which are mental attitudes of the Sender;
- $\texttt{B}_{\texttt{agent}}\phi$ and $\texttt{I}_{\texttt{agent}}\phi$ respectively mean that agent *believes* that ϕ is the case, or that agent *intends* to reach a situation where ϕ is the case;
- $\texttt{Done}(\alpha)$ means that α has "just" taken place.[4]

In our approach, for the sake of heterogeneity, we do not want to model mental states. We rather distinguish between (a) interactions as observable facts, once they take place (at a society/institution level, external perspective), and

[3] As we already had the chance to mention, the use of a specific mapping is an open issue for discussion (see example from [Vas98]). Other mappings could be possible, e.g., we could use the FIPA inform communicative act, of which we give an example:

```
(inform
  :sender (agent-identifier :name i)
  :receiver (set (agent-identifier :name j))
  :content
     "weather (today, raining)"
  :language Prolog)
```

The reason why we use request and not inform, besides its being more intuitive, is that we want to specify a commitment in the dialogue move semantics, based on the type of dialogue move, and independent of its content. In other words, we would rather use inform to make an utterance that does not imply any commitment ("*weather (today, raining)*"), and request otherwise. The dialogue moves accept and refuse can be mapped into FIPA's accept-proposal and reject-proposal.

[4] From [FIP01]: "Done (α, p) means that α has just taken place and p was true just before that", and "Done (α) $\equiv$ Done (α, True)"

(b) dialogue moves as actions that agents plan to execute in order to make an intention feasible (internal perspective).

From an external perspective, communicative actions (dialogue moves) are (i) constrained by permitted interaction protocols and (ii) followed by *commitments*, while from an internal perspective dialogue moves are *motivated* by the desired effect of them.[5] Our external view of agent interactions is similar to that of [Sin98], and it shares with such work the interest in systems that are truly heterogeneous, where it seems unreasonable to assume that agents can predicate on other agents' mental states[6].

Therefore, the semantics of communicative actions can be expressed in a similar way to the one of FIPA, but substituting the internal perspective described by mental states with an external perspective described by the commitments that a certain action brings about.[7] In the case of `request`, a first option could be the following:

$$
\begin{array}{l}
< \texttt{Sender}, \texttt{request}(\texttt{Receiver}, \texttt{give}(\texttt{R})) > \\
\quad \texttt{Commit} : \texttt{accept}_{\texttt{Receiver}}\texttt{request}(\texttt{give}(\texttt{R})) \\
\qquad \rightarrow \texttt{accept}(\texttt{R}) \\
\quad \texttt{RE} : \texttt{gives}(\texttt{Receiver}, \texttt{Sender}, \texttt{R})
\end{array}
$$

meaning that: (i) if `Sender` makes a request about a resource R to `Receiver`, then he commits to accept R from `Receiver` in case `Receiver` accepts the request, and (ii) the purpose of this act is to obtain R from `Receiver`.

We put together two aspects: one is social the other is internal. The choice whether to specify or not the rational effect in the CCL semantics is not trivial. After all, if an agent makes a request about a resource, the purpose of the action could be not to achieve the resource, but – for instance – just to make another agent waste time (it is the case of denial-of-service attacks). By expliciting the *rational effect* on the communicative act, we again refer to something that is internal to the agent and that in an open society cannot often be guessed, nor verified. On the other hand, it could be useful to include the concept of rational or desired effect of a communicative act in the agent model: some agents will

[5] Again, although this work focuses on the interactions and not on the individuals, this strictly depends on how computees are internally modelled.

[6] "Agents may have to have beliefs and intentions, be able to plan and perform logical inferences, or be rational. These constraints also preclude many practical agent designs because you cannot uniquely determine an agents mental state." [Sin98]. In [YS02a] the authors propose a formalism (named *commitment machines*) to express protocols, and show how commitment machines can be compiled into a finite state machine for efficient execution, and prove soundness and completeness of such compilation procedure.

[7] In doing this, we follow an approach that is very much related to that of [FC02], where the authors introduce a social notion of commitment and an object-oriented operational paradigm for it, in order to define a meaning of some speech acts such as *propose*, *accept*, and *reject*.

require resources because they need them, others will require resources because they want to test some other agent's patience.

If we decide to drop the rational effect from the semantics, a semantics of **N** expressed purely in terms of commitments could be the following:

$< \texttt{Sender}, \texttt{request}(\texttt{Receiver}, \texttt{give}(\texttt{R})) >$
$\quad \texttt{Commit} : \texttt{accept}_{\texttt{Receiver}}\texttt{request}(\texttt{give}(\texttt{R}))$
$\qquad \rightarrow \texttt{accept}(\texttt{R}, \texttt{Sender})$

$< \texttt{Sender}, \texttt{accept\_request}(\texttt{Receiver}, \texttt{give}(\texttt{R})) >$
$\quad \texttt{Commit} : \texttt{give}(\texttt{R}, \texttt{Receiver})$

$< \texttt{Sender}, \texttt{refuse\_request}(\texttt{Receiver}, \texttt{give}(\texttt{R})) >$
$\quad \texttt{Commit} : \emptyset$

Such commitments that are implied by communicative acts could be again mapped into integrity constraints, as shown in Figure 3.

$tell(X,Y,request(give(R)),D,T) \wedge$
$tell(Y,X,accept(give(R)),D,T') \ \wedge \ T' > T$
$\rightarrow \ commit(X,Y,accept(R),T'') \wedge$
$\quad commit(Y,X,give(R),T'') \ \wedge \ T'' > T$

Fig. 3. Integrity constraints expressing *commitments*

We imagine that in this society/institutions there are certain types of commitments that govern the agent interactions. In particular, an agent X can commit to accept a resource from an agent Y, or it can commit to give a resource to an agent Y. In our model we also presume the possibility to express conditional commitments (commit to do α once ϕ becomes true, e.g., $\phi = \texttt{accept}_{\texttt{Receiver}}\texttt{request}(\texttt{give}(\texttt{R}))$).

The commitments can therefore be thought at two different levels: at the *society* level in terms of the constraints that implement them, and at the *CCL language* level in association with some specific communicative acts.

6 Discussion

This report proposed a layered architecture for agent interactions where the society defines the allowed interaction protocols, and the protocols are defined by means of constraints. The semantics of communicative acts is given in terms of commitments. In order to explain the ideas, we adopted throughout the paper as a running example a resource exchange scenario.

Relevant work to this purpose is also represented by work of [HdBvdHM99] where Hindriks et al. propose a logic-based approach to agent communication and negotiation where deduction is used to derive information from a received message, and abduction is used to obtain proposals in reply to requests.

The authors propose two pairs of communication primitives: `ask`/`tell` and `req`/`offer`. The first pair are used for information exchange, the second pair are used to communicate requests between agents. Deduction and abduction are used to give semantics to the communication primitives. In particular, deduction serves to derive information from a received message. Abduction serves to obtain proposals in reply to requests. A semantic based on deduction is proposed for the `ask` and `tell` primitives, similarly to other proposals in the literature, while a semantic based on abduction is proposed for the `req` and `offer` primitives.[8]

In our preliminary work on communication via abduction [GMT+02], we propose a computees communication framework where the exchange of knowledge is mapped on hypotheses to be abduced by communicating agents. As a matter of further investigation, we are interested in studying if the proposed CCL can be easily mapped into this framework.

In work by Fornara and Colombetti [FC02], the authors define an operational semantics of main FIPA ACL primitives by means of *commitments* expressed as objects, possibly updated by specific methods. It is worth to be investigated whether the proposed CCL language based on constraints and abducibles is able to also recover this kind of *commitments*.

We are aware that this area has been intensively investigated, and it is not in the aims of this report to survey all the related work present in literature. Although this is only a preliminary work, we believe that an approach to modeling agent/computee interactions by mean of a computational logics-based formalism has several major advantages. It is based on a declarative representation and therefore it could be easy to understand, it is not too far from a potential operational model, which could be used to achieve an implementation of societies of computees based on their formal specifications, and finally, due to this link between formal specification and implementation, it provides a good ground for verification and formal proof of properties.

Acknowledgements

Special thanks to Cristiano Castelfranchi for providing an overview and many interesting pointers about agent society modelling, and in particular for the

[8] "The *communicative action* `offer`(a, ϕ) is seen as advertising that the agent is prepared to offer ϕ to agent a. [...] alternatively, the agent offers a *proposal* ϕ as a way to achieve a request of agent a. The proposal ϕ may contain *free variables*, which indicate the range of proposals the agent is prepared to make. That is, by offering ϕ, any instance of the free variables in ϕ is offered. The process involved in the receiving agents' part is that of abduction. The receiving agent attempts to *abduce* suitable instances θ for the free variables in ϕ such that, given the current agent's beliefs σ, the request ϕ of an agent a is entailed by the proposal, i.e. $\sigma \cup \phi\theta \models \phi$. Moreover, the computed proposal $\phi\theta$ must be consistent with the agent's beliefs σ, i.e. $\sigma \not\models \neg\phi\theta$"

classification of Section 3. This work is the result of a series of discussions that took place during the SOCS meetings of the current year (2002). This work was partially supported by "Progetto Giovani Ricercatori" of the University of Bologna, and by the SOCS project, funded by the CEC, contract IST-2001-32530. The information provided is the sole responsibility of the authors and does not reflect the Community's opinion. The Community is not responsible for any use that might be made of data appearing in this publication.

References

[CFV02] Marco Colombetti, Nicoletta Fornara, and Mario Verdicchio. The role of institutions in multiagent systems. In *Proceedings of the Workshop on Knowledge based and reasoning agents, VIII Convegno AI*IA 2002, Siena, Italy*, 2002.

[CLMT02] A. Ciampolini, E. Lamma, P. Mello, and P. Torroni. An abductive logic programming architecture for negotiating agents. In *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA)*, volume 2424 of *LNCS*, pages 14–26. Springer Verlag, September 2002.

[EdlCS02] M. Esteva, D. de la Cruz, and C. Sierra. ISLANDER: an electronic institutions editor. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002), Bologna, Italy*, Bologna, Italy, July 15-19 2002. ACM.

[FC02] N. Fornara and M. Colombetti. Operational specification of a commitment-based agent communication language. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002). Part II, Bologna, Italy*, pages 535–542. ACM, 2002.

[FIP01] FIPA Communicative Act Library Specification. Experimental specification XC00037H, Foundation for Intelligent Physical Agents, August 2001. Published on August 10th, 2001, available for download from the FIPA website: `http://www.fipa.org`.

[GMT+02] Marco Gavanelli, Paola Mello, Paolo Torroni, Evelina Lamma, and Fabrizio Riguzzi. An abductive approach to communication. Technical report, DEIS, University of Bologna, 2002.

[HdBvdHM99] K. Hindriks, F. de Boer, W. van der Hoek, and J.J. Meyer. Semantics of communicating agents based on deduction and abduction. In *Foundations And Applications Of Collective Agent Based Systems (CABS)*, 1999.

[NS02] P. Noriega and C. Sierra. nstitutions in perspective: An extended abstract. In *Sixth International Workshop CIA-2002 on Cooperative Information Agents*, volume 2446 of *Lecture Notes in Artificial Intelligence*. Springer, 2002.

[Sin98] M. Singh. Agent communication language: rethinking the principles. *IEEE Computer*, pages 40–47, December 1998.

[SOC] SOCS: Societies Of ComputeeS (SOCS): a computational logic model for the description, analysis and verification of global and open societies of heterogeneous computees. SOCS Home Page URL: `http://lia.deis.unibo.it/Research/SOCS/`.

[STT02] F. Sadri, F. Toni, and P. Torroni. An abductive logic programming architecture for negotiating agents. In *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA)*, volume 2424 of *LNCS*, pages 419–431. Springer Verlag, September 2002.

[Vas98] V. Vasudevan. Comparing agent communication languages. http://www.objs.com/agility/tech-reports/9807-comparing-ACLs.html, 1998.

[WJK99] M. Wooldridge, N. Jennings, and D. Kinny. A methodology for agent-oriented analysis and design. In *Proceedings of ACM Agents'99*, 1999.

[YS02a] P. Yolum and M. Singh. Commitment machines. In *Intelligent Agents VIII, 8th International Workshop, ATAL 2001 Seattle, WA, USA, August 1-3, 2001 Revised Papers*, volume 2333 of *LNAI*, pages 235–247. Springer Verlag, 2002.

[YS02b] P. Yolum and M.P. Singh. Flexible protocol specification and execution: applying event calculus planning using commitments. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002), Part II, Bologna, Italy*, pages 527–534, Bologna, Italy, July 15-19 2002. ACM.