E. Verharen and F. Dignum. Cooperative information agents and communication. In M. Klusch and P. Kandzia, editors, *Proceedings of the first international workshop on cooperative information agents, LNAI-1202*, pages 195--209, Kiel, Germany, 1997. Springer-Verlag.

# COOPERATIVE INFORMATION AGENTS AND COMMUNICATION

E. Verharen

Infolab, Tilburg University
PO.Box 90153, 5000 LE Tilburg, The Netherlands
tel. +31 13 4662767, fax. +31 13 4663069, email: E.M.Verharen@kub.nl

F. Dignum

Faculty of Mathematics & Computer Science
Eindhoven University of Technology
PO.Box 513, 5600 MB Eindhoven, The Netherlands
tel. +31 40 2473705, fax. +31 40 2463992, email: dignum@win.tue.nl

## Abstract

Research in Information Systems has switched its focus from data to communication. The communication between different autonomous Information Systems requires a certain amount of intelligence of each system. The system should be able to know which queries it can/may handle and also be able to negotiate about the information that it will give. In short, these systems evolve into what is called Cooperative Information Agents. We describe an architecture for these CIA's in which the relations of a CIA with other CIA's are handled on two levels. The messages themselves are handled by the communication manager. The communication manager can also negotiate a contract with other CIA's. The contracts (which may include communication or transaction protocols) between agents are handled by the contract manager of the CIA responsible for the contract. The messages between the agents are modeled using speech act theory which provides for a rich and flexible communication. In addition, we describe a lexicon in which the conceptual meaning of the terms of communication can be defined. Together, these levels provide an integrated and rich semantics for the communication between CIAs. These can be interorganizational, as in EDI applications, or intraorganizational, as in Workflow Management.

**keywords:** speech acts, deontic logic, lexicon

# COOPERATIVE INFORMATION AGENTS AND COMMUNICATION


## Abstract

Research in Information Systems has switched its focus from data to communication. The communication between different autonomous Information Systems requires a certain amount of intelligence of each system. The system should be able to know which queries it can/may handle and also be able to negotiate about the information that it will give. In short, these systems evolve into what is called Cooperative Information Agents. We describe an architecture for these CIA's in which the relations of a CIA with other CIA's are handled on two levels. The messages themselves are handled by the communication manager. The communication manager can also negotiate a contract with other CIA's. The contracts (which may include communication or transaction protocols) between agents are handled by the contract manager of the CIA responsible for the contract. The messages between the agents are modeled using speech act theory which provides for a rich and flexible communication. In addition, we describe a lexicon in which the conceptual meaning of the terms of communication can be defined. Together, these levels provide an integrated and rich semantics for the communication between CIAs. These can be interorganizational, as in EDI applications, or intraorganizational, as in Workflow Management.


## 1 Introduction

Traditionally an information system (IS) was considered as one central database and a set of users accessing the database through application programs or directly via an SQL interface. Today, databases are connected to each other and have to be accessible using electronic networks and EDI, while still maintaining their autonomy. Complete integration of the various resources might not be possible for technical or organizational reasons, hence the growing reliance on interaction between systems. This lead to the paradigm of cooperative information systems introduced in [15]. Furthermore, for systems to be able to cooperate with others they must have an intelligent interface that can cope with all types of requests and eventualities. In this light CIS's become active in several ways. A CIS actively maintains its information; it can communicate with other systems and reason about the information that it contains. It might decide to search for information that it needs by inquiring for it from other CIS's if it knows that it does not contain the information itself, preferably in ways it negotiates with (and lays down in contracts with) those other systems. It can respond more intelligently to messages explaining why a request does not have an answer, or propose alternatives. And it can negotiate about which requests it is willing to respond to and which requests will have no effect. For this purpose the CIS should contain a task module that plans the tasks the CIS has to fulfill. We refer to an autonomous CIS with tasks and contracts as a *Cooperative Information Agent* (CIA). This term was also used by Klusch in [10] and in the title of this workshop.

In our view the CIA approach has consequences for what is called database semantics and knowledge representation. The traditional focus on logic and algebra to describe the semantics of database or knowledge base content needs to be widened to include the semantics of the communication. Since interfaces often have to reconcile different conflicting viewpoints, and have to be established by different, autonomous, parties, and because they are therefore more difficult to adapt, there is an increasing need for standards, such as reference

ontologies. In [10] this problem is tackled by introducing Local Information Models (LIM) for each CIA. These are more linguistic based descriptions of the content of the data or knowledge base that is maintained by the CIA. In a way these LIM's are the database interfaces through which the other CIA's can approach the database. In our approach we place much more emphasis on the communication and negotiation between CIA's that takes place in occasional contacts. Because the abilities of the CIA to communicate and negotiate take an important place we claim that the influence of *linguistics* for these systems should go beyond that of a natural language interface.

In this paper, we introduce a (conceptual) architecture for CIA's, that integrates much of our previous work, that we hope to use as a basis for future implementation. Although it does not play a big role in this paper, the linguistic theory of Functional Grammar [7], including its theory of the Lexicon, plays a crucial role in the organization of the information towards the other CIA's. One could say that the LIM's of our CIA's are based on this theory. We use the theory of speech acts as developed by Searle [19, 20] and Habermas [9] to describe the communication itself.

Because a CIA must be able to reason about its tasks and the information that is possesses we think it is crucial that there is an underlying formal theory in which the agents can be described (including the communication). We have described this theory in a multi-modal logic (see e.g. [6]). In this paper, because of space limitations, we only indicate how the communication part can be formally described in this logic.

The structure of this paper is as follows: in section 2, our framework of a Cooperative Information Agent is presented. In section 3 we describe the task manager which is the central unit of the CIA in more detail. In section 4 we describe the function of the Lexicon for communication and the role of Functional Grammar. Section 5 takes a closer look at the communication components of a CIA, i.e. the communication manager and contracts. Because of space limitations we only briefly hint on the formal specification technique. Section 6 compares our perspective with other agent approaches, and section 7 gives some conclusions and areas for further research.

## 2 CIA Framework

In figure 1 we show the (global) architecture of a CIA as we use it. In the rest of this section we will give a short overview of the most important features of the components in this architecture.

An *agent* is typically a piece of software that has certain capabilities, actions that it can perform, and certain tasks and responsibilities (delegated to it by a human agent). The actions can be database actions such as updates, or communicative, such as providing some piece of information, or sending a message to another agent. Each agent also has an *agenda* containing the actions to be performed by the agent, instantly or at some designated time. In a normative system this agenda consists of the *obligations* of the agent. An obligation is the result of a commitment to perform a certain act and authorizations restrain or allow the commitment to and operation of an act (including doing other communicative acts). We assume that the agenda is not fixed but can be manipulated by the agent. The agent can add new obligations to the agenda (typically done on the request of another agent) and can reason about them. Obligations can be the result of the (sub)task of the agent, but can also follow from the contract (see section 4). Items can be removed from the agenda by performing actions or by violating the obligation. In the latter case usually some compensatory action has to be performed.

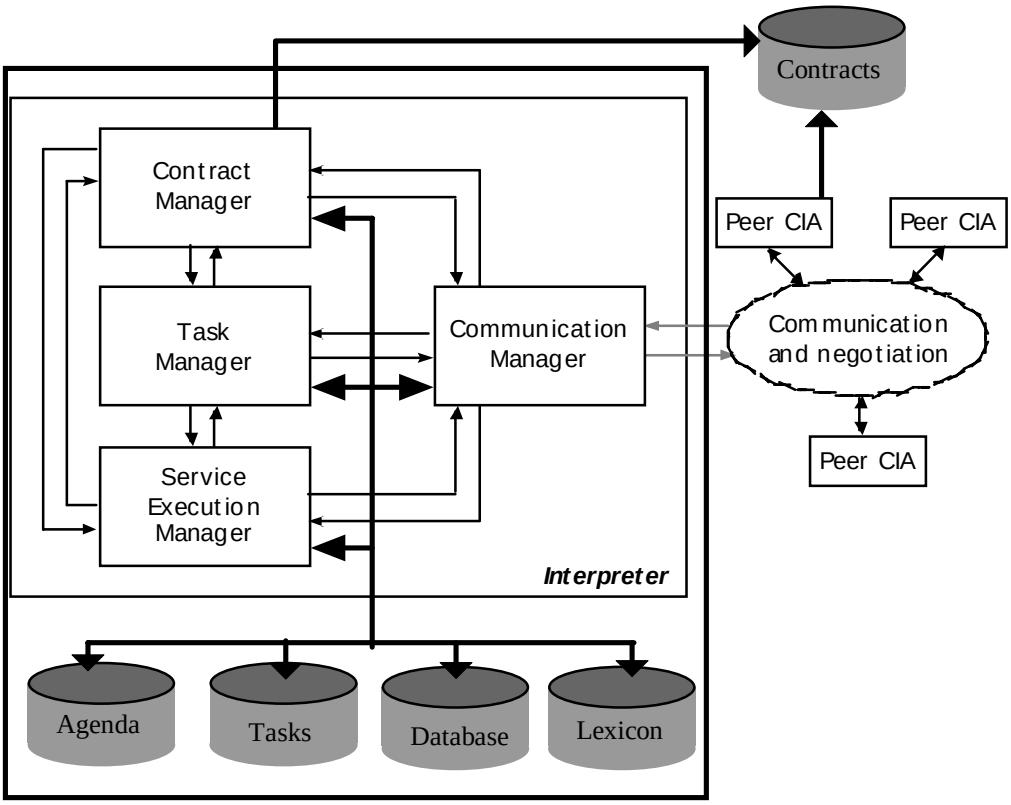


**figure 1. CIA architecture**

The principle engine of an agent is its *task manager*. The task manager maintains the tasks that an agent (can) perform(s). That is, it plans and schedules the tasks, it maintains the agenda and it handles failures of tasks. The pro-active behavior of a CIA is determined by the tasks that the CIA gets at its inception. New tasks can arise from the interactions of the agent with other agents. On the lowest level the agent performs *actions* and *transactions*. Actions can be performed by the agent itself without interaction with other agents. Transactions are actions that involve the interaction with other agents. The transactions are stored in a transaction database that is not shown in the above figure. The basic building blocks of transactions are the messages which will be described in section 5. The tasks and the task manager are described in the next section.

Transactions are used in a *contract* describing the communication behavior between two agents concerning some business relation and process. The contract also specifies what should happen in case of violation of one of the obligations or cancellation by one of the agents, possibly leading to other obligations described by another transaction in the contract and triggering a “contingency” plan, describing what should happen in order to get the subtask fulfilled. The *contract manager* monitors the contracts that a CIA is involved in and decides what steps to take when a contract is breached. Contracts and contract managers are more fully described in section 4.

The third functional component in the CIA architecture is the *communication manager*. The communication manager handles all external communication of a CIA. We do not assume that the CIA’s follow a fixed communication protocol. Therefore we need a rich communication language in which also the intent of each message is clear. This is achieved by basing the messages on the theory of speech acts (Searle [19]). Using these speech acts we can model existing protocols that are often used like contract net. However, the CIA can also react when other agents do not follow the same protocol. The communication manager draws heavily on the *Lexicon* for the definition of the terms that are used in the communication with other CIA’s.

The Lexicon is the system that stores and manages the terminology of a certain domain, it describes the information of the discourse including the message types, and corresponds to the Conceptual Model describing the static and dynamic structure of the Universe of Discourse. The population of this model (possibly including inference rules) is stored in the database. We will say more about the lexicon in the next section.

The last functional component of a CIA is the *service manager*. The service manager handles the services that a CIA can give to other CIA's. That is, it checks whether the other agents are authorized to request the service and if the service is available. It may also handle simple exceptions within the services (usually when the service can be restarted). The service manager also maintains a database with information about services that the CIA can provide itself and services provided by other CIA's. In this way it assists the task manager at the execution of tasks that the CIA cannot perform itself but has to ask from other CIA's. We do not describe the service manager in this paper as it is of little importance for the communication of CIA's.

The only component we did not mention so far is the actual database or knowledge base that is maintained by the CIA. We do not make any presupposition about the format of the database. It can be a relational database, an object oriented database, a deductive database, just a flat file or anything else containing information. The service manager functions as an interface between the database and the rest of the CIA. It is possible to use the theory of Functional Grammar to describe the information as is shown in [24]. It gives a uniform representation for the information, the lexicon and the communication. We will not describe it any further here because the use of FG is not crucial to the architecture of the CIA.

# 3 Tasks and the task manager

## 3.1 Tasks

A *task* is a meaningful unit of work assigned to an agent. Performing the task often involves initiating communication transactions (This can be seen as (1) in figure 4, section 5 where the contracts and communication are described). However, the task's specification and updates thereof do concern the agent in question only, whereas changes in the possible transactions (involving other agents) can only be made by consent of the other agent. For that reason, we have made a distinction between task and contract, where the contract corresponds to the agreements between the two agents and the task draws on this potential for fulfilling an agent's goal.

Tasks are typically described in some task language (e.g. TasL [13]). It typically allows the specification of tasks and subtasks, alternatives and temporal constraints. Especially when tasks involve the initiation of transactions they are prone to many types of failures, including traditional ones like system failures (in case of interoperable transactions also including network failures) and possible deadlocks caused by concurrent processes ([12]). More interesting a failure occurs when a transaction (as one subtask) is initiated that does not commit (e.g. request info) ((2) in figure 4), but also when the transaction does commit first, and the resulting deontic state is violated later (e.g. the company doesn't deliver), or because of cancellation ('cancel', see figure 4), whereby the other party undoes an achieved effect. These features of transactions directly influence the task specification. For a CIA, it is important that tasks are not embedded in application code, but made explicit so that the Reasoner can use them in scheduling or rescheduling the work. Rescheduling is necessary when a subtask fails or a planned subtask becomes superfluous.

### 3.2 The task manager

The execution model of the Task Manager will be as follows: when a task is called or a goal is established, the Task Manager devises a plan to fulfill the goal or perform the task. The plan consists of a number of subtasks with some precedence relations and alternatives. The subtasks are put on the agenda in the right order. The task manager then tries to perform all subtasks in the right order, backtracking when a task fails or an exception occurs. To keep the task specification simple, we give the designer the opportunity to specify a contingency plan separately (see below) (triggered by (4)).

Our execution model enforces a "structured approach" in the task specification. We do not allow for arbitrary abort or commit dependencies between subtasks over the boundaries of the parent tasks. This modularity is enforced to keep the specification transparent and maintainable. In [6] a first attempt in implementing tasks is made by giving preferences to goals. This also gives us the opportunity to reason over preference relations on (sub)tasks and how they influence each other.

A notorious problem with contingencies is that later (dependent) subtasks may already have completed, but their result may have become obsolete. Whether they have to be retried or not depends on what kinds of results they have produced. We therefore give the designer the opportunity to specify a separate contingency plan. The contingency plan consists basically of a set of *results* (such as supplier list, ticket-reservation). Results have an internal object structure (not specified here) and come about by certain subtasks and can become invalidated by other subtasks. When this occurs, a task can be triggered to repair the damage. This task can make use of the fact that all the essential results obtained so far (and not invalidated) are explicit. For instance, in the case where a hotel-reservation is dependent on the airline-reservation, and the flight gets canceled; the contingency plan can try to repair the damage by trying another airline. Only if that fails the hotel-reservation has to be canceled (independently, there can be a sanction on the party canceling, as specified in the contract and monitored by the contract manager).

## 4 Lexicon

In this section we discuss the Lexicon as the place to store (linguistically) specified organization knowledge.

The Lexicon is the system that stores and manages the terminology of a certain domain.. It explicitly represents the *mutual* understanding about a certain concept of all agents involved. This can be used in the initial phase (negotiation) of setting up the contract. Besides this, there is a ‘private’ part that contains the agent’s knowledge about the domain.

In [23] a linguistic approach is presented in which the lexical definition as well as the lexical structures are based on linguistic primitives. The Lexicon defines the possible predicate frames describing the domain. That is, structural information, in particular the taxonomic relation, designating subsumption or subtyping, but also semantic sets and pre- and postconditions for dynamic predicates; and conceptual information (predicate schemata, i.e. ‘stereotypes’, including essential but not necessary characteristics of a concept), specified in Functional Grammar.

**Note** that lexical definitions should not be considered as exhaustive characterizations of a concept. A definition is made relative to a certain context. Within the context, the definitions must differentiate different concepts, and provide a *basis* for mutual understanding.

Following linguistic theory, the concepts are organized in a taxonomy and around prototypes. The higher levels of this taxonomy form a basic ontology, including primitive concepts like "entity", "event", "state", "sign". In the case of complex concepts, such as actions, the lexical entry includes a frame containing the roles of the participants, such as "agent", "recipient", etc. The definition of a concept is in principle simply the (natural

language) dictionary entry. However, the definition can be parsed and stored internally in a linguistic representation formalism like FG, which allows formal treatment. Moreover, lexical research has shown that definitions typically make use of a small array of basic structures, such as the "isa" and "partof" relation and "purpose". This allows for even more formalization and hence computational processing, while the definition can still be expressed in natural language.

Following are a few examples of lexicon definitions. The italicised noun phrase indicates a taxonomic link to a superconcept. The arguments are given between parentheses:

```
car: a closed road vehicle on more than three wheels driven by a motor engine
model(car): the name of a registered car design
registration no.(car): a string of 6 alphanumeric characters uniquely
identifying a car
```

The following example defines the action **sell** as a transfer action with three semantic roles (the object has an empty label, `ag` stands for agent and `rec` for recipient). The incondition says that the action includes a payment action of the recipient.

```
sell(ag X:human)(P:thing)(rec Y:human)
isa transfer
pre = own(ag X)(P), price(P) = D
post = own(ag Y)(P)
inc = pay(ag Y)(D)(rec X)
```

In circumscribing the terminology for a particular application domain, the knowledge engineer might draw on available terminologies for the generic domain. Such generic terminologies might draw in turn on more general dictionaries. We assume that the CIA's can draw their own lexicon from this hierarchy of lexicons. When more ontologies, ISO standards and domain lexicons become available, it is possible for negotiating CIAs to set up a mutual understanding by making a reference to such a given set of concepts. It could also use a bilingual (certified) Lexicon that translates Dutch business concepts to French, for example. It should be clear to which definitions both CIAs commit themselves .

## 5 Communication and Contracts

In general we can distinguish four phases of communication. The first phase is the inquiry and negotiation about the terms of the contract. In this phase authorizations can be established on the basis of which some actions can be performed in the following phases.

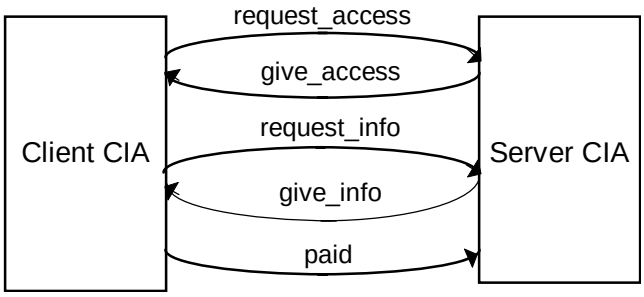


**figure 3. Communication in a (prototypical) information gathering procedure**

In figure 3 this corresponds to the requesting and sending of access rights. The access rights authorize the client to request some information on some specified conditions. In most of the literature about the interactions between autonomous systems this negotiation phase gets most attention. In [18] protocols are given according to which the negotiation should be performed. Many theories from game theory are used to secure an optimal outcome of the

negotiations and coalition forming (see e.g. [10]). In this architecture we abstract away from the considerations (utility functions) that determine the behavior of each CIA. Each CIA can use a different criteria to enter a cooperation. We only describe the effects of the messages that the agents send during the negotiation. In this way we can describe exactly which are the obligations and authorizations under the contract resulting from the negotiations.

The second phase consists of the acceptance of the contract, i.e. the authorizations and obligations that follow from the contract. In the above example this is included in the request which implies an acceptance of the conditions of the access rights. The third phase is the fulfillment of the contract, i.e. following a protocol according to the terms agreed upon in the contract. In the example this corresponds to the actual information request, information delivery and payment sequence. What makes our CIA's different from most other cooperating systems described in the literature is that also in this phase the CIA's are autonomous. They can at any moment decide not to honor the contract they agreed upon. Of course this will have repercussions, but a CIA might decide they are less damaging to him than following the contract. So, even though a CIA agreed upon disclosing some information to another agent it might decide later to withhold some of it because it is too secret.

The last phase is the satisfaction phase, which has no specific messages in the model. This is perhaps because satisfaction is often implicit, and only dissatisfaction leads to communicative acts, such as appeals to warrant.

### 5.1 Communication modeling

To describe the semantics of communication processes, we need to specify the meaning of the *information* and *communication* aspects of the messages. The specification of the structure of the *message* is based on the theory of speech acts as developed by Searle ([19]). Illocutionary logic ([20]) is a logical formalisation of the theory and is used to formally describe the message structure itself, that is, the types and effects of the messages. The illocution (=illocutionary force) of a speech act indicates what the intention of the speech act is. E.g. the speaker might be asserting, denying, predicting, confirming, greeting, baptizing, etc. The illocution of the message first of all defines the illocutionary type, and a propositional context (both of which can be expressed in FG structures).

The basic illocutionary types that we use are ASSERT, DIRECT, COMMIT and DECLARE ([1, 19, 20]), but many more can be distinguished (cf. [2, 3]. For a motivation of the set of basic speech acts we refer to [4]).

The DIRECT is used to give orders or requests to other agents. E.g. "give me all information about all managers earning more than $200.000". This type of message can result in an obligation for the receiving agent to deliver the information. The obligation only arises if there is some basis of authorization for the request (or order). The authorization can be given explicitly beforehand in a negotiation phase or it can follow from a hierarchical relationship between the agents (one is the "boss' of the other). Important is that the effect of the message depends on the relationship between the agents. Also the effect of a DIRECT message is at most an obligation. The receiving agent can still autonomously decide to fulfill the obligation or not.

The COMMIT is used to create obligations for oneself. If a request is made by another agent which has no authorization then an agent can honor the request by committing itself to the action. E.g. "I will give all information about all managers earning more than $200.000 and less than $300.000".

The ASSERT is used to inform another agent about a (believe about a) fact. E.g. "the salary of Carl Boss is $100.000". It is, of course, up to the receiving agent to determine whether it believes this fact as well. Again if prior to this message the sending agent has gotten some authority to base the speech act on then the receiving agent might have to believe the fact.

The last illocution is the DECLARE. This type of messages are the only ones that can actually create new (abstract) facts. E.g. “I give you access rights to the employee database”. The effect of this message is that the receiving agent now has some new rights. Of course only if the message is based on some authority. As can be seen from the example, this type of message can be used to explicitly give authorizations to other agents. It is therefore an essential message type for the negotiation phase while it hardly occurs in the other phases of the communication.

We argue that with the above basic illocution types all types of primitive messages that are used in other protocols, like e.g. contract net, can be described. We have shown this for some common message types such as OFFER, COUNTEROFFER and ACCEPT in [14].

It is a characteristic of messages that they seldom stand on their own. For example, a request is typically followed by an acknowledgment, commitment or refuse message. Therefore messages can be organized in transactions. This is done on basis of the effect of one message and the preconditions of the next message. In this way protocols can be build for sequences of messages that appear often. However, the communication is not limited to protocols that are predefined.

### 5.2 Dynamic Deontic Logic

Because the messages have effects for other agents and the agents must be able to reason about these effects it is very important that the messages and their effects are formally described. For the semantics of communication models, we have made use of Dynamic Deontic Logic ([11, 25]). The fundamental reason for the use of deontic concepts in communication modeling is that coordination of behavior always requires some form of mutual commitment. If an agent does not execute an action he has committed himself to, this causes a violation of the contract. Because the action should be executed in the future, it cannot be guaranteed, so the interpretation “it will happen in all future courses of events” is too strong, but the interpretation “it will happen in some course of events” is too weak. Interpreting “$\alpha$ is obligatory” as: “not doing $\alpha$ violates a commitment” we get a more precise meaning of what it is that something is on an agent's agenda. Violations do not cause logical inconsistency, but can be the trigger for sanctions or repair actions.

The Deontic Dynamic Logic is combined with Illocutionary logic to obtain a full logical framework in which the communicative behavior of a CIA can be specified. At this place we only give a brief description of the different types of messages and their effects in this logic. For a full description of the formal semantic models and language we refer to [4,5, 25].

The main component of the following formulas is every time of the form $[\alpha]\varphi$, which means that after the performance of the action $\alpha$ the formula $\varphi$ holds.

The COMMIT is described with the following formula:

$$\mathbf{[COMMIT(i,k,\alpha)][DECL(k,P_{ik}(\alpha(i)))]O_{ik}\alpha}$$

Which means that after agent “i” commits itself towards agent “k” to perform “$\alpha$” and agent “k” declared that agent “i” is permitted to perform “$\alpha$” then agent “i” has the obligation towards “k” to actually perform “$\alpha$”.

The effect of a DIRECT depends on the authorization of the speaker:

$$\mathbf{auth(i,DIR(i,k,\alpha)) \rightarrow [DIR(i,k,\alpha)]O_{ki}\alpha}$$

Which means that if agent “i” is authorized then after it orders agent “k” to perform “$\alpha$” agent “k” has the obligation to perform “$\alpha$”.

$$\mathbf{[DIR(i,k,\alpha)]K_k\ INT_i\alpha(k)}$$

If agent “i” is not authorized the only effect of the order is that agent “k” knows that agent “i” intents him to perform “$\alpha$”.

The effect of a DECLARE also depends on the authorization of the speaker:

$$\mathbf{auth(i,DECL(i,f)) \rightarrow [DECL(i,f)]f}$$

If agent "i" is authorized then the declaration of the fact "f" actually creates that fact.

$$\boldsymbol{\varphi \rightarrow [DECL(i,f)]Pref_i(f \mid \varphi)}$$

If agent "i" is not authorized the effect of the declaration of the fact "f" under the circumstances φ is only that agent "I" prefers the fact "f" to hold in the circumstances that φ hold.

Finally the effects of the ASSERT are described as follows in logic:

$$\mathbf{[ASS(i,k,f)]K_k\ B_i f}$$

If agent "i" asserts the fact "f" to agent "k" then afterwards agent "k" knows that agent "i' believes fact "f".

$$\mathbf{auth(i,ASS(i,k,f)) \rightarrow [ASS(i,k,f)]B_k f}$$

If agent "i" asserts the fact "f" to agent "k" with authorization then afterwards agent "k" also believes fact "f".

With the above formulas we can infer the exact effects of each message in a protocol. Both in the case when a certain message is expected (and thus usually authorized) as well when it is unexpected. It is also possible to calculate the precise effects of the complete communication protocol that is used. This makes it easier to react to breakdowns in the communication.

### 5.3 Contracts

The agreements about the interactions between the CIA's are described in the contracts. In figure 4 we give an overview of the relation between the contract and the tasks of the CIA's involved in the contract.

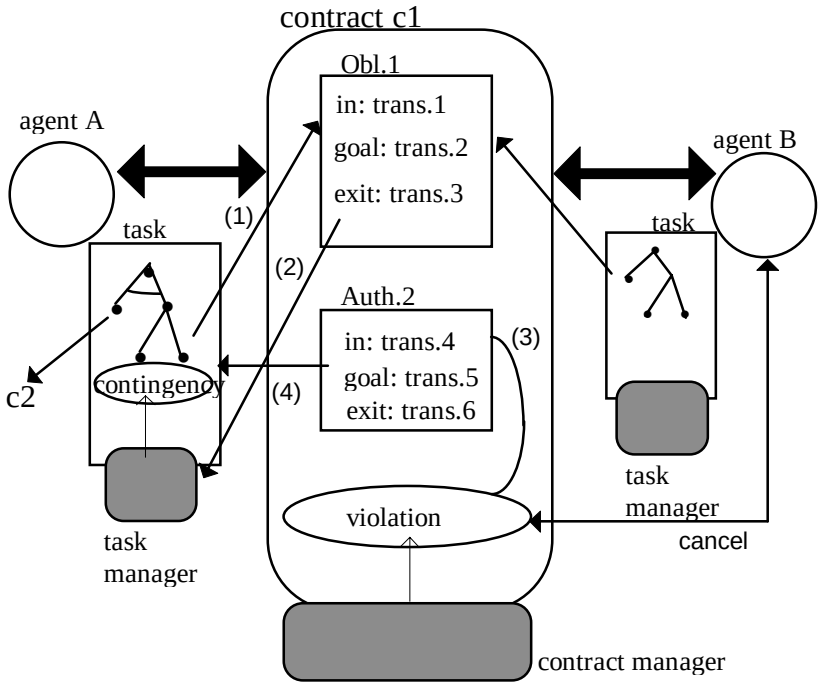


**figure 4. Contracts and Tasks**

In our framework, contracts conceptually specify obligations between different parties about services provided to each other. If a particular service is not being fulfilled, it is possible to reason over this violation and take a remedial action without forcing the whole task to abort ((3) in figure 4). A contract describes the authorized communication behavior among providers and receivers of services. If the provider does not adhere to the obligation, in the case of a

CIA, it is the job of the Contract Manager to impose the violation policies. It would complicate the task specifications and lower the reusability if this communication is included in the task. As stated before the Task Manager should only be responsible for ensuring that a task is brought to its goal, not how violation of commitments are dealt with. In this way, the process is more flexibly responsive to failures.

From a technical point of view, a contract is nothing but a protocol binding different parties to their commitments by explicitly specifying the type of services agreed upon, the obligations and the failure recovery methods. From an organizational point of view, a contract between interoperable systems stemming from different organizations also has the purpose of laying down some agreement. By grounding it in other contracts or business law, it may have legal status. Contracts, in the general sense of agreements between commercial partners, can be more or less elaborate. For instance, in Goldkuhl's model [8] a contract is the result of the negotiation, and boils down to a commitment to deliver and a commitment to pay. We assume that such a contract is set up by the two partners only once and then frequently used.

Since contracts can be more or less elaborate, we should require from the communication modeling approach that it can account for different levels of contracts. We briefly want to say how this can be done in our CIA approach. Starting from an empty cyberspace in which agents can just send messages, agents must have the possibility to offer services (described by their individual tasks). A supplier agent can then provide the service of giving an offer. When a customer agent requests for this service, the supplier can refuse or accept. If he accepts, he sends an offer, that is, an authorization for the customer to order a product with the obligation to pay for it. If the customer agent uses this authorization, an agreement has been accomplished. All that is required in this case is a reliable and standardized way of requesting a service, as well as the “umbrella” protection of some international business law that guarantees the legal status of the obligations of both supplier and customer described in the contract. Nowadays it is often considered an economic advantage when business relations can be tied closer with some preferred supplier or customer. In that case, the two parties want more agreements beforehand, for example about a guaranteed delivery time. Such a contract has to be set up. This can be supported by another service of the agent that offers contracts (that can be provided by the organization) rather than specific products. If the customer accepts the offer, a mutually agreed contract is accomplished.

## 6 Comparison with Other Agent Frameworks

In this paper we introduce an agent-oriented framework for CIAs. Agent theories and architectures have gained much interest lately, both from the software engineering and AI discipline.

Although there is no consensus in the AI community about precisely which combination of information attitudes (like knowledge and belief) and pro-attitudes (like desire, intention, commitment, choice) are best suited to characterize agents, a number of approaches have gained much support. Perhaps the best known agent theory is the BDI framework of Rao and Georgeff ([16]).

Rao and Georgeff developed a logical framework on three primitive modalities: Belief, Desires and Intention. And although in other work ([17]) they consider the potential for adding (social) plans to their formalism, the BDI framework considers agents in isolation; it ignores the communicative aspects that are central in our approach.

Another influential approach is the one taken by Shoham ([21]) in which he uses the mentalistic notions knowledge, belief, intention, and obligation to characterize an agent. He also describes a new programming paradigm based on this notion of agents: agent-oriented programming. The CIA framework takes a similar stance to agent characterization, as

described in section 2 and above, in which the obligations play a central role. In [22] we described how we can use this paradigm to experiment with an implementation of the CIA framework.

The architecture that probably resembles ours most closely is the one used in the ADEPT system (see [14]). They also specify services and contracts. However, the communication protocols that the agents can use are more limited than ours. Also they do not have a specific component dealing with ontology's. An agent must keep so-called acquaintance models for all agents that it communicates with to determine the terminology that it can use.

In [10], Klusch describes an architecture of CIA's that do have a specific component that deals with common ontology's etc. He specifies a Local Information Model for each agent through which other agents can see the information that the agent contains. The CIA's defined here, however, lack an explicit communication component. The communication protocols are fixed. They do however give more explicit details about the ways optimal contracts (coalitions) can be formed. However, once a contract is established the CIA's cannot violate the contract anymore, thus loosing some of their autonomy.

## 7 Concluding Remarks

In this paper, we have described an architecture for Cooperative Information Agents. We propose the use of CIAs in different application settings, such as Business Computing, EDI, and Workflow Management.

Contracts were presented as a way of modularizing the normative specifications. Both the negotiation phase to establish the contract as well as the contract itself can be modeled using our formalism. Contracts were specified by using interoperable transactions, together with deontic constraints and rules for appropriate action when violated.

The use of a rich communication language makes it possible to specify very flexible interactions between agents. By using contracts, which establish obligations and authorizations between agents it is possible to minimize the communication that is needed in standard situations.

In the lexicon we can describe the ontology used by the agent. Although we did not describe it in this paper, the agreement of the agents about the meaning of the terms that are used in the communication is of prime importance. This can be supported by using standard lexicons for general terms and related domain lexicons for specific domain knowledge.

We have sorted out the complex concept of 'compensation' into two more focused notions: compensation of the other party, as specified in the contract, and compensation, or contingency handling, of the task. For contingency handling, we propose a separate specification part that monitors the results obtained and invalidated so far. Although exception handling is a complex issue and will remain so, the least we can do is try to manage the complexity. This is the most important but modest goal of the task/contract distinction. We also consider it an advantage that task management can be turned into a local issue, rather than a concern of the (global) multidatabase. In this way the global control is kept to a minimum, which makes the specification and implementation more flexible.

Our main short-term goal is to build a prototype implementation of CIA's based on the architecture that we sketched in this paper and test our ideas in practice. We are currently working on an implementation using Java as implementation platform.

## 7 References


1. Austin J.L., *How to do things with words*, Clarendon Press, Oxford, 1962.

2. Balmer T. and W. Brennenstuhl *Speech Act Classification: A Study in the Lexical Analysis of English Speech Activity Verbs*, Springer-Verlag, Berlin, 1981.

3. Chang M.K. and C.C. Woo, “A Speech Act Based Negotiation Protocol: Design, Implementation and Test Use”, *ACM Transactions on Information Systems (TOIS)*, vol.12, no.4, pp. 360-382, 1994.

4. Dignum F. and H. Weigand, “Communication and Deontic Logic”, *Information Systems, Correctness and Reusability*, R. Wieringa and R. Feenstra (eds.), World Scientific, Singapore, pp. 242-260, 1995.

5. Dignum F. and H. Weigand, "Modeling Communication between Cooperative Systems", *Proc. of CAISE'95*, J. Iivari et. al. (eds.) (LNCS-932), Springer-Verlag, Berlin, pp. 140-153, 1995.

6. Dignum F. and B. van Linder, "Modeling Rational Agents in a Dynamic Environment: Putting Humpty Dumpty Together Again", *Proc. of the 2nd workshop of the ModelAge Project*, J.L. Fiadeiro and P.-Y. Schobbens (eds.), Sesimbra, Portugal, January 1996, pp. 81-92, 1996.

7. Dik S.C., *Functional Grammar*, North-Holland, Amsterdam, 1978.

8. Goldkuhl G., “Information as Action and Communication”, *The Infological Equation, Essays in honor of B. Langefors*, B. Dahlbohm (ed.), Gothenburg Studies in Information Systems, Gothenburg Univ., 1995. (Also: Lingkoping Univ. Report LiTH-IDA-R-95-09)

9. Habermas J., *The Theory of Communicative Action: Reason and the Rationalization of Society, Volume One*, Beacon Press, Boston, 1984.

10. Klusch M. and Shehory O., “Coalition Formation Among Rational Information Agents”, W. vd Velde and J. Perram (eds), Proc. MAAMAW-96, LNAI-1038, Springer-Verlag, p. 204-217, 1996.

11. Meyer J.-J.Ch., “A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic”, *Notre Dame Journal of Formal Logic* 29(1), pp. 109-136, 1988.

12. Ngu A., R.A. Meersman and H. Weigand, “Specification and verification of communication for interoperable transactions”, *Int.l. Journal of Intelligent and Cooperative Information Systems* 3(1), pp.47-65, 1994.

13. Nodine M.H., N. Nakos and S. Zdonik, “Specifying Flexible Tasks in a Multidatabase”, *ACM SIGOIS Bulletin* 16 (1), p. 13-17, 1994.

14. Norman T., Jennings N., Faratin P. and Mamdani E., “Designing and Implementing a Multi-Agent Architecture for business process management” J. Mueller, M. Wooldridge and N. Jennings (eds.) Intelligent Agents III - Proceedings ATAL-96, p.149-162, Budapest, Hungary, 1996.

15. Papazoglou M. et.al. “An organizational framework for intelligent cooperative IS”, IJICIS-1(1), 1992.

16. Rao A.S. and M.P. Georgeff, “Modeling rational agents within a BDI-architecture”, *Proc. of Knowledge Representation and Reasoning (KR&R-91)*, R. Fikes and E. Sandewall (eds.), Morgan Kaufmann Publishers, San Mateo, pp. 473-484, 1991.

17. Rao A.S. and M.P. Georgeff, “Social plans: preliminary report”, *Proc. of Decentralized AI 3 - MAAMAW-91*, E. Werner and Y. Demazeau (eds.), Elsevier Science Publishers, Amsterdam, 1992.

18. Rosenschein J. and Zlotkin G., “Rules of Encounter”, MIT Press, 1994.

19. Searle J.R., *Speech Acts*, Cambridge University Press, 1969.

20. Searle J.R. and D. Vanderveken, *Foundations of illocutionary logic*, Cambridge University Press, 1985.

21. Shoham Y., “Agent-oriented programming”, *Artificial Intelligence* 60, pp. 51-92, 1993.

22. Verharen E. H. Weigand, and O. De Troyer, “Agent-oriented information system design”, *Working Papers. of ISCORE’94*, R. Wieringa and R. Feenstra, (eds), Vrije Universiteit Report IR-357, Amsterdam, pp. 378-392, 1994.

23. Weigand H., *Linguistically Motivated Principles of Knowledge Based Systems*, Foris, Dordrecht, 1990.

24. Weigand H., “Assessing Functional Grammar for Knowledge Representation”, *Data and Knowledge Engineering* 8 (1992), pp. 191-203, 1992.

25. Weigand H., E. Verharen, and F. Dignum, "Integrated Semantics for Information and Communication Systems", *Proc. of IFIP DS-6* , R. Meersman, L. Mark (eds), Stone-Mountain, Georgia, IFIP, 1995.