Introduction

Frank Dignum
Mark Greaves

1 Technology for Agent Communication

Agent technology is an exciting and important new way to create complex software systems. Agents blend many of the traditional properties of AI programs - knowledge-level reasoning, flexibility, pro-activeness, goal-directedness, and so forth - with insights gained from distributed software engineering, machine learning, negotiation and teamwork theory, and the social sciences. An important part of the agent approach is the principle that agents (like humans) can function more effectively in groups that are characterized by cooperation and division of labor. Agent programs are designed to autonomously collaborate with each other in order to satisfy both their internal goals and the shared external demands generated by virtue of their participation in agent societies. This type of collaboration depends on a sophisticated system of interagent communication. The assumption that interagent communication is best handled through the explicit use of an agent communication language (ACL) underlies each of the papers in this collection. In this introduction, we will supply a brief background and introduction to the main topics in agent communication.

Formally, ACLs are high-level languages whose primitives and structures are expressly tailored to support the kinds of collaboration, negotiation, and information transfer required in multi-agent interaction. ACLs exist in a logical layer above transport protocols such as TCP/IP, HTTP, or IIOP. The latter deal with communication issues at the level of data and message transport, while ACLs address communication on the intentional and social level. ACLs themselves are complex structures composed out of different sublanguages that specify the message content, interpretation parameters such as the sender and the ontology, the propositional attitude under which the receiver should interpret the message content, and several other components. Typical ACLs also have a characteristic mentalistic semantics that is far more complex than standard distributed object protocols. This means that ACL design is a delicate balance between the communicative needs of the agent with the ability of receivers to compute (in tractable time) the intended meaning of the message. Further, it is important that the syntax, semantics, and pragmatics of the various components of an ACL be as precise and explicit as possible, so that the agent systems using that ACL can be as open and accessible to developers beyond the original group.

This last point bears some emphasis. Historically, many multi-agent systems have been built using somewhat *ad-hoc* and developer-private communication mechanisms. [1] Although these systems often contain many independent agents and can exhibit impressive accomplishments, the agents involved often rely on a large number of communicative assumptions that are not true of arbitrary agent collections. These assumptions range from the presumption of a shared ontology and problem domain to specific nonstandard meanings for messages (or the absence of a message) that are tailored to particular contexts. These often-undocumented assumptions are made by agent developers for reasons of communication efficiency or developer convenience, and knowledge of them is critical to properly interpret the agent message traffic in these systems. So, while such purpose-built agent collections are important to test and validate different hypotheses and approaches to agent problems, they can be extremely difficult to generalize and extend without extensive interaction with the original developers. The locus of this problem can be traced to these implicit domain-specific assumptions in the agent communication design.

The papers in this collection all address a set of issues in general ACL and agent interaction design. The ACLs that they discuss are intended to have explicit principles surrounding their proper interpretation in a context of use.[2] Further, these ACLs are also designed to be generally applicable to a wide variety of agent interaction types. The combination of explicitness and generality leads to extremely expressive languages with well-defined semantics that are grounded in powerful logics. Thus, these ACLs demand a great deal from the agents that must interpret them: computing the meaning of an arbitrary message may require extensive deductive machinery, and is typically formally intractable. However, using a strong ACL with a consistent semantics that can be precisely known in advance is a great advantage when creating heterogeneous agent systems that are designed to be easily extendible and broadly useful.

The first ACL that was designed to be both standard and general came as a consequence of the DARPA Knowledge Sharing Initiative (KSE). The Knowledge Query and Manipulation Language (KQML) [5] is the subject of several papers in this collection, and was originally devised as a means for exchanging information between different knowledge-based systems. However, because of the generality of its high level primitives and its message orientated structure, KQML was

[1] Agents can also communicate with actions other than classically linguistic productions, simply by making observable changes in the environment that have semantic force. For example, an agent that locks a resource for itself might be assumed to communicating to an observer its need for the resource at that time. However, without a general semantic and pragmatic theory of action, it is impossible for other agents to precisely characterize the meaning of such actions, or to understand them as communicative. And, the provision of such a theory simply turns the applicable actions into a *de facto* communication system, albeit one with an unorthodox syntax. For this reason, the papers in this volume concentrate on communication with ACLs.

[2] Although the ACLs described in this volume are very generic, they all make certain assumptions about the agents that use them and/or the environment in which they are used. It is not possible to create an ACL without any assumptions at all; hence the emphasis on making these assumptions explicit.

found to also function well as language for agent communication. KQML is the most widely implemented and used ACL in the agents community.

More recently, an agent standards body called the Foundation for Intelligent Physical Agents (FIPA) has proposed its own ACL, FIPA-ACL.[3] FIPA-ACL was designed to remedy some of the perceived weaknesses of various versions of KQML, and to be a suitable language for the agents community to standardize upon. FIPA-ACL differs from KQML in several ways; some of the more significant ones are the following:

- FIPA-ACL was derived from Arcol [7], and includes a precise semantic model based on a quantified multimodal logic. This semantics makes it possible to describe an agent's communicative acts and their intended effects in an extremely precise way. However, FIPA-ACL's semantic model is so powerful that a FIPA agent cannot in general deduce the intentions of the sending agent from a message received from that agent; thus, agents using FIPA-ACL are forced to employ a variety of semantic simplification strategies (such as conversation polices) to aid in message interpretation. KQML, on the other hand, has a relatively simpler semantics, and can be substantially easier to use in applications where the domain is reasonably restricted.
- KQML includes a number of special administrative and networking message types to make it easier for agent system developers to support registration, brokerage, directory services, streaming data, and the like. In FIPA-ACL these administrative functions have to be performed using the normal REQUEST and INFORM messages and specialized content.
- Because FIPA-ACL is defined and administered by a standards body, the actual ACL is relatively slow to change, and each addition is carefully vetted. KQML has no central administrative body, and so KQML users have developed several incompatible "dialects" by adding individually specialized message types and tinkering with the basic semantics of KQML primitives.

Most of the papers in this collection are based on either KQML or FIPA-ACL.

Finally, we will point out an important area of agent communication and ACL use that the papers in this volume do not emphasize. Successfully using an ACL to communicate between two agent programs is dependent on the proper functioning of a great deal of non-ACL communications infrastructure. This infrastructure involves message ordering and delivery, formatting and addressing, directory services, gateway-style translation, quality-of-service, and other standard networking and communications issues. In practice, implemented agent systems have used both centralized strategies that handle all aspects of messaging between agents (often implemented in KQML through the introduction of a "facilitator" agent), as well as more decentralized systems that devolve this functionality to the communication handlers of the agents themselves (as is typically

[3] FIPA's name for its ACL is technically just "ACL," but to avoid confusion we will refer to it here as "FIPA-ACL."

done in FIPA-ACL). Agent systems exhibit little standardization in this area. Further, implementations of ACLs often expose more of the details of this infrastructure than one might strictly like. The safest thing to say is that there are many deep and difficult problems when designing a general agent communications infrastructure that will be efficient and reliable for different communication topologies and networking systems, and the papers in this volume do not address them except in passing.

2 Issues in Agent Communication

In this section we will introduce a set of issues that have been important in the development of ACLs and agent communication theory generally. Although we will use KQML and FIPA-ACL as examples and point out a number of potential problems with them, we do not intend to criticize the effort put into the design of these languages. Much of our knowledge of these problems came from the practical experience of researchers applying KQML or FIPA-ACL in their systems, and we are indebted to the many researchers who have devoted enormous efforts to these ACLs.

2.1 Theories of Agency

One of the core research issues in the agent communication community involves the linkage between the semantic underpinnings of the ACL and the theory of agency that regulates and defines the agent's behavior. In order for the messages of an ACL to be formally coherent, these two theories must be aligned.

A theory of agency is a general formal model that specifies what actions an agent can or should perform in various situations. Like the Turing model of computability, it abstracts away from any particular implementation, and functions as a normative theory that is useful for analysis. Theories of agency for software agents are usually based on a small set of primitives derived from the propositional attitudes of philosophy (*e.g.,* belief, knowledge, desire, and intention) and a set of axioms or axiom schema which define their entailment relations. A complete theory of agency also includes accounts of the agent's general reasoning strategy and deductive model, its theory of action and causality, its account of planning and goal satisfaction, its system of belief dynamics and revision, and so forth. An agent need not directly implement its theory of agency, but it must behave as if it did. Examples of the elements which compose a theory of agency include Moore's accounts of knowledge and action, Rao and Georgeff's BDI architectures, Singh's know-how and branching time systems, Cohen and Levesque's intention theories, and so forth. Different agent systems will combine different elements to comprise their own theory of agency.

An agent's communicative behavior is among the behaviors regulated by a theory of agency. Because of this, the semantic theories that define the meaning of an ACL message must ultimately be linked to the entities provided by the agent's baseline theory of agency. Current versions of both KQML and FIPA-ACL handle the linkage between the semantic theory and the theory of agency by appealing to a simplified version of natural language speech act theory (originally developed in Searle [8] and Grice [3]). In this approach, agent communication is treated as a type of action that affects the world in the same way that physical acts affect the world. Communicative acts are performed in service of intentions, just like any other act. Unlike physical acts, however, the primary effect of a communicative act is to change the beliefs of the parties involved in the communication.[4]

The current semantic theory of FIPA-ACL depends on a theory of agency which supplies a set of BDI-style primitives. The semantics of FIPA-ACL is based on mentalistic notions such as belief and intention, and (because of its speech-act theory component) treats agent messaging as a type of action. Formally, this means that FIPA-ACL's semantic theory is expressed in an extremely powerful quantified multimodal logic involving both belief and intention as primitive operators, as well as a simple theory of action. As a result, agents that aspire to use FIPA-ACL in a semantically coherent way are required to adhere to a BDI-style theory of agency. They also face the somewhat daunting task of acting as if they implemented a reasoning engine for the semantic account.

In contrast to the FIPA-ACL, KQML did not originally assume a full BDI architecture of the agents. Rather, the original KQML semantics were defined in terms of a very simple theory of agency centered on adding and deleting assertions from a virtual knowledge base. The assumptions made about the required behavior of KQML agents were very weak, and the resultant semantics of KQML messages were much more permissive than that of FIPA-ACL. As is now well know, this permissiveness allowed wide latitude in KQML implementations, and the contributed to the proliferation of different and incompatible KQML dialects. Labrou's [5] second-generation semantics for KQML was much more precise, and based on a sophisticated BDI-style theory of agency similar to that of FIPA-ACL. However, Labrou's use of modal logic to specify the preconditions, postconditions, and completion conditions for each KQML communicative act type made the complexity of semantic reasoning for KQML messages comparable to that required by FIPA-ACL.

Mismatches between the theory of agency and the semantic theory can occur when the theory of agency licenses communicative actions that are not expressible in the ACL semantics. The *sincerity* condition on agent ACL usage is one such example. Sophisticated theories of agency often allow agents to act with

[4] Certain special classes of acts, parallel to the *explicit performatives* of natural language, can also in some circumstances directly change the properties of objects in the world. For example, a priest can marry a couple by simply pronouncing them husband and wife. In the agent world, there are examples of this when agents register themselves with a matchmaker or broker, or when they grant other agents access rights on a database they manage.

the intent to deceive if it furthers the agent's goals. This is often cited as a requirement for optimal behavior in electronic commerce applications and adversarial negotiations generally; for example, the revenue-maximizing strategy of an agent might involve deceiving another agent about the first agent's true valuation of a good. However, in order to make the message semantics as useful as possible, most ACL semantic theories (such as the KQML and FIPA-ACL theories) require that agents never use that ACL to assert something that they do not themselves believe. This is a strengthening of the analogous principle for humans: we do not typically assume that our interlocutors are lying to us. But it also makes possible the situation that an agent might desire to communicate something that its ACL cannot legally express.

The sincerity condition serves as a simplifying assumption for agent communication. Another such assumption involves the ability of an agent to reliably observe the effects of another agent's actions. Applied to agent communication, this is often taken to mean that the interagent communication channels are error-free. Agent systems routinely assume that all messages eventually arrive to their intended recipients and are not distorted by the environment (or malicious actors) in transit. Often, it is further assumed that the order of the messages that are sent to the same destination does not change during the transportation. Depending on the agent's execution context, these assumptions may not be appropriate. Again, the possibility exists that simplifying assumptions in the ACL could foreclose certain types of desirable or rational behavior relative to the agent's theory of agency.

2.2 ACL Semantics

In order to introduce our discussion of ACL semantics, it will be useful to first look at the simpler domain of programming language semantics. A common method of specifying the semantics of actions in programming languages involves defining the preconditions and postconditions of the corresponding statements. For example, the semantics of $x = 2.1$ might include the precondition that x should be the name of a location able to store a value of type real. The postcondition of this action would be that the value stored in the location denoted by x is 2.1. The precondition of the action is designed to enforce the condition that the action will only be attempted in meaningful situations. So, in case x is an integer variable it will not be possible to assign the real number 2.1 to it. The postcondition tries to capture a kind of minimal effect of the action. The relation between the pre- and postcondition correlates to the intuitive meaning of the action. In the case above, whenever x is the name of a location that can store values of type real, then whatever the value is stored in that location, after the action $x = 2.1$ the value 2.1 will be stored in that location.

The pre- and postconditions of actions in programming languages can be expressed in terms of variables and their values before and after the action, because the relevant types of actions are limited to manipulating the values of variables.

However, communicative acts in an ACL do not directly manipulate variables and their values. They are conceived to operate at the higher level of abstraction given by the theory of agency, and refer to the primitives supplied by this theory. Therefore, the preconditions and postconditions for communicative acts are typically expressed in terms of the mental attitudes of the involved agents. For example, the precondition of KQML's TELL message states that the sender believes what it tells and that it knows that the receiver wants to know that the sender believes it. The postcondition of sending the TELL message is that the receiver can conclude that the sender believes the content of the message.[5] The semantics for FIPA-ACL is based on a similar semantic approach that involves specifying a message's feasible preconditions and rationally expected effects.

Although the precondition/postcondition approach can supply a minimal meaning for messages in an ACL, situations occur frequently where it is desirable to overload this minimal meaning with a more precise and context-specific gloss. This leads to a tension in ACL semantic theory. On one hand, we want the semantics to be flexible enough to be applicable in all situations where agents use the ACL. Therefore we formulate very general pre- and postconditions in the formal statement of the semantics. On the other hand, the resulting pre- and postconditions are often so abstract that they are not fully adequate in all situations. Furthermore, it is often very difficult to verify whether the agent's state in fact satisfies the pre- and postconditions. This is partly due to the fact that, although we routinely ascribe mental attitudes to agents, agents are almost never actually programmed using these concepts directly. For example, how would one verify, for agents α and β and a proposition ϕ, that "α knows that β wants to know that α believes ϕ?"

A second distinction between the semantics of agent communication and that of programming languages is the fact that the agents are distributed and (ideally) autonomous pieces of software. For an ordinary program the postcondition of an action can be precisely defined and enforced, because the complete program context is available and the individual elements of the program are not autonomous relative to one another. However, in a multi-agent system the specific postconditions of an action cannot be enforced. This is apparent in the distinction between the actual effect and the intended effect of a speech act for an agent. When an agent tells some information to another agent, it typically does so with the intention that the other agent will at least come to believe that the sending agent believes that information. However, because agents are autonomous, an agent can never *directly* change the beliefs of another agent, and so the effect of a speech act cannot be guaranteed. Agents do not have complete control over the actual effect (in speech act terminology, the *perlocutionary effect*) of their communication.

[5] KQML also has the notion of a completion condition, which roughly corresponds to the conditions that obtain after the successful performance of the act in a normal and cooperative communicative context.

A final, related point involves the intersection of ACL semantics with the complex social relationships of obligation and commitment that constrain the actions an agent may take. Most of the time, an agent does not just shout in the dark; agents typically perform communicative acts with the expectation of a particular reaction from the receiving agent(s). An agent's communicative behavior is the result of implicit or explicit planning by that agent. For example, a request for information is usually sent with the expectation that the receiving agent will supply the requested information, or will respond to the requesting agent that it does not have the information or cannot supply the information. Conversely, an order to perform an action will be sent with the expectation that the receiving agent will perform the ordered action. So, the only reason for an agent to include the performance of a REQUEST in a plan is if it concludes that the response to the REQUEST will further the goals of the plan. At some level, this is a precondition to the use of REQUEST, however this type of pragmatically-inspired precondition is extremely difficult to express in traditional ACL semantic theory. Obligations on the receiving agent are similarly difficult to express.

The above discussion highlights the social aspect of agent communication. This is a facet of agent communication that is only beginning to be addressed. Agent designers have usually assumed that the networks of obligation and power relationships that characterize human social behavior are not relevant to multi-agent systems. In practice, however, idiosyncratic social conventions have ended up being embedded in agent architectures and interaction protocols, with the result that different agent systems exhibit significant incompatibilites in this area. More research is needed into characterizing these fundamental communicative concepts in a multi-agent systems context. This includes concepts such as "commitment," "obligation," "convention," "power" (in the sense of hierarchical relations), and so forth. Once these concepts are clarified, it then becomes possible to build a unified ACL semantics and pragmatics that takes account of these concepts.

2.3 Ontologies

An issue that is closely related to ACL semantics is the proper treatment of ontologies in an ACL. Both FIPA-ACL and KQML include an element that is used to identify the source of the vocabulary used in the message content. This is designed to make these ACLs independent of particular application vocabularies, and to give the message recipient a way to interpret the nonlogical terms in the message content. In the original specification of KQML from the KSE, this element was designed to refer to an ontology specified in Ontolingua. In FIPA-ACL, the semantics of the ontology tag is effectively user-defined.

The way that an agent would make use of the KQML or FIPA-ACL ontology specifier to interpret unfamiliar parts of an ACL message has never been precisely defined. Merely supplying an ontology tag does not solve the problem of how agents acquire and use the common ontological knowledge base that is a prerequisite for successful communication. This is a particularly acute problem in open systems that include agents based in different organizations. The problems associated with learning meanings and reasoning with a new set of terminology are very similar to those in the area of database integration and cooperative information systems: somehow the ontologies that the different agents use have to be "integrated." Of course, ontological integration does not mean that the terminological structures have to actually be unified, but at minimum there must exist translation rules that convert relevant terms from one ontology into the other. Although a human working with representatives of each terminological community can often hash out a satisfactory set of rules, it is almost impossible to construct these rules fully automatically ([6, 4]). Consequently, agents can only fully communicate if they already share a common ontology, or if a set of preexisting translation rules is provided.

Although this may seem very restrictive, it has not been so disastrous in reality. For example, standards for product descriptions are very common in trade groups. And, in many open systems the agents communicate initially through some third party that initiates the contact between the agents. This third party will often mandate an ontology that all agents will use, and ontologies thus mandated will typically be built in to the agents by their developers. This is the case in most electronic auctions, where selling agents include specialized code to specify their product using predefined terminology. Nevertheless, the general ontological problem is still the subject of active research.

2.4 Completeness of ACL Message Types

An issue that is closely related to the theory of agency concerns the completeness of the set of message types in the ACL. Because ACLs can be used in arbitrary communicative contexts, one important goal is that their basic set of message types be sufficient to express all possible kinds of agent communicative intent that are allowed by the underlyingi theory of agency. Without a complete message set, agents and their developers may find themselves in situations where they are forced to invent additional *ad hoc* meanings for certain ACL messages, with the attendant decline in interoperability.

An interesting non-agents case of this is the evolution of the X.12 EDI standard. X.12 included a very precise and complete syntactic description of its messages, but had only an informal natural-language description for the message semantics. X.12 began with a limited set of messages that worked in most cases, but has been repeatedly extended over the years when a situation occurred that could not be intuitively captured by simple combinations of the existing message types. Thus, X.12 currently contains message types with overlapping meanings, odd patterns of expressivity, and various other logical problems. This practical but unprincipled approach to X.12 development has given rise to a message set for which it is very difficult, even for humans, to select the correct message type to use in a particular case.

The evolution of KQML and FIPA-ACL has also been involved questions about the completeness of the provided message types. KQML was designed by a loose group of researchers, and its message types were driven by the goals and needs of DARPA's KSE program. The designers of FIPA-ACL patterned their language after KQML, and because they were working in the framework of a formal standards body, they created several processes designed to limit the sorts of problems found with X.12. However, in both ACLs the stock of officially-sanctioned communicative acts is limited to several varieties of directives (*e.g.,* INFORM or TELL) and assertives (*e.g.,* REQUEST or ASK). This means that each ACL is incapable of expressing all agent intentions that are possible in powerful theories of agency, because several classes of performatives (see Searle [8]) are absent from both. For example, neither ACL can directly express agent commitment, because neither includes commissives like PROMISE. Expressives like WISH also are missing from the standard set of communicative acts supplied in ACLs like KQML or FIPA- ACL. The practical effect of these omissions is limited because both KQML and FIPA-ACL are extensible ACLs: users are free to invent new application-specific performatives as long as they do not overlap or clash with the standard set. However, this freedom encourages the development of different incompatible dialects of these ACLs.

2.5 Conversation Policies

In the preceding sections we have discussed ACL research issues that are primarily related to the generation and interpretation of individual ACL messages. A final topic we will address is how to bridge the gap between these individual messages and the extended message sequences, or *conversations*, that arise between agents. As part of its program code, every agent must implement tractable decision procedures that allow that agent to select and produce ACL messages that are appropriate to its intentions. This is not purely a problem of matching ACL semantics to agent intention: except in the most limited of agent systems, these decision procedures must also take into consideration the context of prior ACL messages and other agent events. Paradoxically, taking this context into account can actually *simplify* the computational complexity of ACL message selection for an agent. By engaging in preplanned or stereotypical conversations, much of the search space of possible agent responses can be eliminated, while still being consistent with the ACL semantics. The specification of these conversations is accomplished via *conversation policies* (CPs).

Because of this computational advantage, virtually all multi-agent systems employ some type of explicit or implicit conversational layer.[6] Theory has lagged practice in this area, however: unlike research in ACL semantics, work on formal

[6] The original KQML specification had only token support for agent conversations; this shortcoming was noticed as early as 1994 in [2] and has been largely corrected using a variety of methods.

accounts of agent conversation remains in its infancy. Terminology and theoretical approaches are still being worked out, formal approaches and metrics are still fairly unsettled, and the role of research in natural language pragmatics and discourse theory is still being evaluated. The wide variety of approaches that are discussed in the papers in the latter parts of this volume testify to the exploratory nature of the research. In this introduction, therefore, we will simply introduce some of the questions that are central to the papers in this collection, and that agents researchers hope to answer in the coming years.

Possibly the overriding theoretical question in the field concerns the linkage between the ACL's semantic theory and its account of conversation. On the one hand, it seems obvious that large-scale properties of agent conversations, such as overall information flow and the establishment of commitments, are a consequence of the individual meanings of the messages that make up the conversation. In this view, the ACL semantics is primary, and every conversational property logically derives from the composition of some collection of semantic properties of the individual messages and their sequence. On the other hand, there is a significant thread of research that takes conversational sequences themselves to be semantically primitive, and the precise meaning of the individual messages is nuanced by their role in the overall conversation. In this view, the conversational semantics are primary, and because of the dependence of ACL semantics on context, the same message might have slightly different meanings when used in the context of different agent conversations. Whether one takes conversational semantics or ACL semantics as logically prior will affect their answers to several definitional questions, such as:

- What exactly is (and is not) a conversation policy? What important properties of agent interaction should CPs capture?
- How are CPs individuated? When are two agent conversations instances of the same policy? Are there interesting equivalence classes of CPs? Is there a useful type hierarchy of CPs?

Once one has settled on a basic theoretical perspective on the linkage between CPs and ACL semantics, then there are still a number of technical questions that remain. For example, there are several non-equivalent candidates for a CP specification language, ranging from transition nets like finite state machines and Petri nets, to various types of logic-based specifications, subgoal trees, and network protocol specification systems. These formalisms vary widely in the degree of conversational rigidity they entail, the models of concurrency they support, the computational complexity of their execution, and the availability of tools and techniques to help verify the analytical properties of the represented conversation policy.

Finally, the use of conversation policies to guide agent communicative behavior engenders a host of practical questions. How should conversation policies be implemented in agent systems? Should CPs be downloaded from a common library, prebuilt into the agent's program, or derived from conversational axioms at runtime? How can conversation policies be negotiated, and unfamiliar policies learned? And finally, how can conversation policies be integrated with the other policies, plans, and rules which define an agent's behavior?

3 About this Volume

As the foregoing has indicated, research in agent communication can be placed along a spectrum of maturity, with research in content languages and network protocols being the most developed, research in ACL semantics being somewhere in the middle, and research on conversation policies and agent social obligations being the most exploratory. The papers in this volume clearly reflect this diversity. In our view, this is one of the most exciting aspects of agents research. The contributions in this book were selected from the workshop on *Specifying and Implementing Conversation Policies*, held in May 1999 jointly with the Autonomous Agents conference in Seattle and chaired by the second author, and the workshop on *Agent Communication Languages: From Speech Acts to Conversations*, held in August 1999 jointly with the International Joint Conference on Artificial Intelligence in Stockholm and chaired by the first author. We decided to join the workshop papers in a single volume because the workshops are very closely related, have different emphasis and together cover most of the research in the field of agent communication.

We have divided the papers into four categories. Section I includes a number of papers concerned with the semantical aspects of agent communication. Section II deals with conversation policy descriptions. Section III concerns itself with fundamental issues surrounding agent conversations. Finally, the papers in section IV consider the relation between agent communication and general agent task planning issues.

Section I: ACL Semantics and Practice Most papers in this describe semantical issues of agent communication. The last paper in this session proposes to use XML as a practical description language for ACLs.

M. Colombetti describes how normative and practical aspects form an integral part of the meaning of agent communication. Although these aspects usually are loosely connected to some of the pre- and postconditions he argues that they also form part of the performance of the speech act (sending the message) itself. He then describes a logic formalism within which the three aspects (semantics, normative aspect and practical aspect) of a message can be described in an integral way. A few consequences for ACL design and conversation protocols are discussed as well.

M. Singh emphasizes the importance of the social aspect of communication in his paper. He uses the theory of Habermas to distinguish different social aspects of communication. With each communicative act correspond three types of validity claims: an objective, a subjective and a practical one. The objective validity claims is based on the truth of the message content. The subjective validity claim is based on the sincerity of the speaker and the practical validity claim is based on the justifiability of the speaker to perform the communicative act. A very important concept in this respect is the social commitment. After a definition of all the concepts in a logic framework he gives some (preliminary) formalizations of the social semantics of speech acts. Finally, some consequences for conversation protocols are discussed.

J. Pitt and A. Mamdani concentrate on one of the validity claims described in Singh's paper. They claim that the sincerity condition that is assumed in all ACLs does not necessarily have to be true at all in non- cooperative settings. They give some examples in which the sincerity condition does not hold in electronic trading. Another important point is that the sincerity condition is hardly verifiable and therefore it is difficult to put it as a precondition on any communicative act. Finally, they show that the claim that agents may be liable to a legal penalty if shown to be non-sincere cannot be sustained. Subsequently they give a new logical formalization of agent communication in which the sincerity condition is not presumed and show the consequences for agent conversations.

Two other papers in this section look at the semantics of ACLs from a different perspective. They use well-known formalisms that were developed for program interaction and see how these formalisms can be applied for agent communication.

K. Hindriks, F. de Boer, W. van der Hoek and J.-J. Meyer use the agent programming language 3APL to describe agents in terms of their beliefs and goals. 3APL is based on a first order language. A number of transition rules describe the operational semantics of an agent. To this formalism two communicative primitives are added (the "ask" and "tell" messages). The semantics of these communicative acts are also described in the form of transition rules within the logic. Logical abduction can be used to obtain proposals in reply to requests, while deduction is used to derive information from received messages. The formalism is illustrated on a multi-stage negotiation protocol.

R. van Eijk, F. de Boer, W. van der Hoek and J.-J. Meyer concentrate on the operational semantics of agent communication. They use a multi-agent programming language based on the well-understood paradigm of object oriented programming. Interaction between the agents takes place by means of a rendezvous mechanism as developed for object-oriented programs. The formal semantics are defined by transition rules for its operational behavior. Because the operational semantics closely follows the syntax of the language, it easily gives rise to an abstract machine to implement it.

The last paper in this section takes a practical view on ACLs and proposes XML as a tool to describe these languages.

B. Grosof and Y. Labrou argue for the use of XML as encoding mechanism for the FIPA ACL. The use of XML facilitates both common parsers for ACLs (synactically linking these ACLs with the WWW world) and also enriches the capability of ACL expressions to link to ontologies. The authors concentrate on a number of business rules that are important as content for the ACLs in electronic commerce settings. These business rules can be encoded in restricted logic programs (called courteous logic programs) that have some desirable properties and can also be described using XML.

Section II: Conversation Policy Description

The papers in this section address general issues with conversation policies.

M. Greaves, H. Holmback, and J. Bradshaw is an introduction to the general notion of an agent conversation policy. The authors first identify the major issue in agent communication for which conversation policies are used, and then use this to derive a set of general requirements for any account of conversation policies. With these requirements in hand, they suggest that the current transition-net based approaches to conversation policy specification are misguided, and argue for a more structured and fine-grained approach. Several desiderata for practical conversation policies are also discussed.

L. Phillips and H. Link start by defining an agent conversation specification, which includes the communicative acts to be used by each party and their sequencing. They then argue for an additional level of policy which would supervene on the conversation specification and include various context- dependent high level rules which govern the specific conversation. They describe how such a two-level mechanism might be implemented, and show how this leads to several beneficial properties between the communicating agents.

S. Moore explores the use of Harel's statecharts to represent different classes of conversation policy. He first describes how conversation policies relate to ACLs and to actual conversations, with an emphasis on a speech-act based ACL called FLBC. He next describes several properties of agent conversation and shows with several examples how statecharts can represent these properties. Finally, he speculates on how conversation policies can evolve and spread through the agent community.

J. Pitt and A. Mamdani show how a different type of ACL semantic theory can ameliorate many of the technical problems faced when trying to define and compose conversation policies. First, they briefly describe how the message semantics of a large class of ACLs can be cast as the intention of the receiver to respond. Using this semantics, they define a small sample ACL, and show how the meanings of the messages in this ACL can be made relative to the context of the conversation in which they occur. They discuss various ways that their sample ACL can be extended, and explore the implications for agent conversations.

Section III: Conversation Specification

The six papers in this section describe specific formalisms that can be used to capture particular policies.

S. Cost, Y. Chen, T. Finin, Y. Labrou and Y. Peng argue that finite state machines are not sufficient to characterize agent interactions that involve concurrency. They propose Colored Petri Nets (CPNs) as an alternative model to capture conversation specification involving concurrency and multiple conversational roles. They describe CPNs and their implementation in Jackal, and provide several detailed examples.

F. Lin, D. Norrie, W. Shen and R. Kremer also discuss CPNs as a useful description and verification tool for conversation policies. They describe a design methodology based on object-oriented design that links higher-level conversation topics with the actual communicative acts in a conversation. This methodology allows several layers of specification. The authors also show how conversation schemata represented by CPNs can be composed, and how agents can use this formalism and to control their conversational behavior.

M. Nodine and A. Unruh describe the conversation policy mechanism implemented by the agents in MCC's InfoSleuth project. InfoSleuth uses carefully crafted finite-state automata to specify the policies. The authors describe several conversational requirements from the InfoSleuth system, and show how their policy formalism meets the requirements. They also show how their system is able to extend and concatenate different conversation policies, and speculate about how to extend their formalism to address more semantically complex types of conversations, such as multicast and delegation.

M. Barbuceanu and W.-K. Lo describe and analyze the conversational coordination language used in the COOL system. The language includes separate notions of conversation plans, conversation rules, and specific execution states. They also address several advanced topics in COOL conversations, including conversation plan composition, the compatibility of conversation plans, group conversations and the application of machine learning technology to support adaptive conversational behavior.

L. Vongkasem and B. Chaib-draa propose to view agent conversations as joint projects (in the same spirit as the paper by R. Elio and A. Haddadi). The conversation is seen as a joint action that has to be coordinated. The agents are put into a social context. They have to figure out not only what is their best action, but also what action (speech act) is expected from them by the other agent(s). The authors conclude that conversations are the basic units of communication and not the speech acts themselves. Therefore, the formal semantics of agent communication should therefore be based on concepts such as joint action, joint intention, etc.

F. Martin, E. Plaza and J. Rodriguez-Aguilar describe their experiences with conversation protocols in IIIA's Fishmarket project. They model the protocols between agents through the use of pushdown transducers, which allows the agents to keep track of nested conversation contexts. The authors also supply a formal analysis of their conversation protocol formalism, concentrating on the notion of protocol compatibility, and discuss issues of conversation protocol negotiation.

Section IV: Conversations and Tasks

One issue in conversation policy design comes from the interaction of agent communicative behavior with the agent's overall goals. How should we treat the relation between conversation policies and agent tasks or plans? The five papers in this section address this issue.

F. Dignum, B. Dunin-Kęplicz and R. Verbrugge start from a linguistic theory of dialogues to describe conversations. A number of rules are formulated in this theory that describe how a conversation can proceed at each point in time. The rules are formalized in a multimodal logic that also describes the mental state of the agents. Therefore a uniform formalism can be used to describe the complete agent interaction. The theory is applied in the context of team formation, where it is clear which type of conditions should hold after each stage. A rough sketch is given how the postconditions of one stage can be reached using the theory and some simple rules for the agents.

S. Noh and P. Gmytrasiewicz discuss the issue of finding an optimal communicative act. They use a setting of mixed human and artificial agents within a simulated air defense domain. The agents use a decision theoretic framework to compute the communicative act with the highest expected utility. The reasoning is based on a probabilistic frame-based knowledge formalism that represents uncertain information the agents have about their environment. The benefits of this communication strategy are measured and compared to the performance of human subjects under the same circumstances.

R. Elio and A. Haddadi describe a two-level framework for relating agent discourse to the problem-solving methods of the agent. By defining a task space and a discourse space, and showing how the two spaces can be linked via agent intentions, they show how conversation policies can be defined by the interplay between conversation protocols and the recognition of the run-time intentions of the agents in a dialogue. They also explore issues in mixed-initiative dialogues.

T. Wagner, B. Benyo, V. Lesser and P. Xuan describe several scenarios involving multi-agent coordination under complex scheduling models, and address the role of conversation policies in this type of situation. They describe how conversation policies can be represented as extended finite-state machines, and how these can be integrated into a general agent coordination toolkit.

C. Jonker, J. Treur and W. Wijngaards describe some possible interactions between verbal and non-verbal communication. They argue that physical and verbal communication both play an important role, and should be treated in combination rather than being seen as disjunct processes within an agent. They describe a model of an agent that interacts with humans, in which the non-verbal communication influences the content of the verbal communication. In the paper, the authors describe a sample implementation and show some parts of it.

References

1. Cohen, P.R. and Levesque, H.J. Communicative Actions for Artificial Agents. In *Proceedings of the International Conference on Multi-Agent Systems* (ICMAS-95), pp. 65-72, 1995.
2. Finin, T., Labrou, Y. and Mayfield, J. KQML as an Agent Communication Language. In Bradshaw, J. (Ed.), *Software Agents*, MIT Press, Cambridge, 1995.
3. Grice, P.H. Logic and Conversation. In P. Cole and J. Morgan (eds.) *Syntax and Semantics, vol.3, Speech Acts*, Academic Press, New York, pp. 41-58.
4. Klusch, M. (ed.) *Intelligent Information Agents.* Springer-Verlag, Berlin, 1999.
5. Labrou, Y. *Semantics for an Agent Communication Language*, Ph.D. thesis, University of Maryland, USA, 1997.
6. Papazoglou, M. and Schlageter, G. (eds). *Cooperative Information Systems: Current Trends and Directions.* Academic Press, 1997.
7. Sadek, M.D. *Attitudes Mentales et Interaction Rationnelle: Vers une Thorie Formelle de la Communication.* Thése de Doctorat Informatique, Universit de Rennes I, France, 1991.
8. Searle, J.R. *Speech Acts.* Cambridge University Press, Cambridge, 1969.