Multi-agent Collaboration: An E-Commerce Example

Abdelaziz Khamis
Dept. of Computer and Information Sciences, Cairo University
Cairo, Egypt
khamis@frcu.eun.eg

Magdy Aboul-Ela
Sadat Academy for Management Sciences
Cairo, Egypt
mmha5_2000@yahoo.com

Mohamed Atwany
Dept. of Computer and Information Sciences, Cairo University
Cairo, Egypt
mohamed.atwany@acm.org

Abstract
E-commerce represents a model of modern commerce in which autonomous software agents represent the roles of different trading entities. Applications of e-commerce are characterized as consisting of a number of distributed heterogeneous entities operating in dynamic environments. In this paper, building on top of an open framework for multi-agent collaboration, we study collaboration between trade entities performing a trade purchase transaction.

Keywords
Multi-agent, Collaboration, Teamwork, Agents, e-commerce.

INTRODUCTION
The term e-commerce stands for a model of modern commerce in which software agents represent the roles of trading entities such as the buyer, the seller, the mediator, the facilitator, and the information provider. The adoption of e-commerce is believed to result into advantages for the enterprises, in the form of cost reduction, and for the customers, by providing them with better bargaining tools [1].
E-commerce applications typically rely on distributed and autonomous activities for information search, fusion, extraction and processing, without centralized control. Business partnerships, as found between suppliers, resellers, brokers, and customers, need to be created dynamically and maintained only for a period of time, an interval of single transaction for example [2].
In this paper, we study the potential for modeling e-commerce entities as a group of agents collaborating to execute a trade purchase transaction. In our research, we build on top of a proposed open multi-agent collaboration framework. In addition, we make use of a proposed framework's implementation that facilitates collaboration within a system of agents.
In the following discussion, we adopt the following definitions. A *software agent* defines a software entity that is able to perceive its environment and is able to act upon it. An agent is able to reproduce itself, and can communicate directly with other agents. An agent possesses skills of its own and can offer services to other agents, and is driven by a set of tendencies. In addition, an agent's autonomous flexible behavior is based on a number of key processes such as problem solving, planning, decision-making, and learning. A *rational agent* is an agent characterized by having all its actions being the result of reasoned deliberation, and is capable of adjusting its means to the ends that it proposes to itself to achieve. A *multi-agent system* (MAS) is a loosely coupled network of agents that work together to solve problems that are beyond each agent's individual capabilities or knowledge. The act of collaboration refers to a forms of high-level cooperation that require a number of interacting agents to develop mutual understanding of the problem and a shared view of the task being solved [3].
This paper is organized as follows. We first introduce an open framework for multi-agent collaboration. Next, we introduce an implementation the for proposed open multi-agent collaboration framework. Then, we introduce the e-commerce case study. Finally, we give summary and conclusion.

A MULTI-AGENT COLLABORATION FRAMEWORK
In this section, we introduce an open framework for facilitating teamwork within a group of possibly heterogeneous rational agents. First, we discuss the need for such collaboration framework. Next, we introduce the framework components.

The Need for an Open Collaboration Framework
In an open multi-agent environment, interaction rules are neither known in advance nor fixed since agents are neither controlled nor identified in advance [4]. This diversity implies that an open multi-agent collaboration framework cannot assume a pre-specified agent architecture, organization or interaction protocol. Such assumptions are considered as constraints on the openness of the collaboration framework.
Current collaboration frameworks like GRATE [5] and STEAM [6] does not support such openness criteria. The GRATE framework enforces a specific agent architecture consisting of two clearly identifiable components, the cooperation and control layer, and the domain layer. In addition, the GRATE agent community enforce a flat organizational structure [5]. The STEAM teamwork model is built on top of the SOAR environment [6].
Our proposed collaboration framework is based on two teamwork models. The first is defined in [7], while the second is defined in [8]. In addition, the proposed framework assumes a layered agent conversational model as defined in [9].
The approach used for developing multi-agent collaboration framework is based on separating high level collaborative behavior, defined as the conceptual model, from possible conceptual model implementations, defined as collaboration framework implementations.

The conceptual layer of the framework consists of (i) a teamwork state model, used to reason about teamwork behavior and (ii) a layered teamwork conversational model, used to enable agents to contribute to teamwork interaction scenarios. The proposed framework implementation layer defines details of messaging, ontology, communication and interaction mechanisms.

Thus, the proposed framework is transparent to different agent architectures, organization, communication and interaction protocols. The proposed approach gives flexibility to the team agent designer to define implementation models in terms of agent communication language, behavior representation language, and message format. For example, an agent designer can develop a framework implementation that adopts KQML (Knowledge Query and Markup Language), KIF (Knowledge Interchange Format) as the agent communication, and agent's behavior representation languages.

Framework Components

The proposed collaboration framework consists of a team work state model and a teamwork conversational model.

Teamwork State Model

The teamwork state model consists of a set of possible team states and a set of possible state transitions. An agent uses this model to reason about the state of the teamwork. In **figure 1**, we outline teamwork states and common transitions between these states, defined as follows:

- **Pre-Planning:** Cooperative problem solving (CPS) starts when an agent, the problem initiator, recognizes the problem. The team then enters into the pre-planning state. At this state, the problem is recognized by candidate team agents, but the conditions that govern team formation have not been satisfied yet.
- **Planning:** At this state, team members create a plan for action. Agents announce their individual and joint commitment to planned sub-actions.
- **Post-Planning:** At this state, team agents have committed to a designated joint action that they mutually believed to by a possible solution to the problem.
- **Execution:** At this state, the team members have jointly intended that designated joint action. The team group or subgroup might jointly intend a joint sub-action of that joint action for execution. In addition, an individual team agent might individually intend actions as part of team group or subgroup intention.

Teamwork Conversational Model

The teamwork conversational model consists of (i) an agent interaction protocol that allow an agent to express its own beliefs, goals, actions, commitments, and intentions, and to query other agents for theirs using the *inform* and *inquire* speech acts, (ii) a set of teamwork interaction patterns covering all possible teamwork interaction scenarios, and (iii) a teamwork conversational policy.

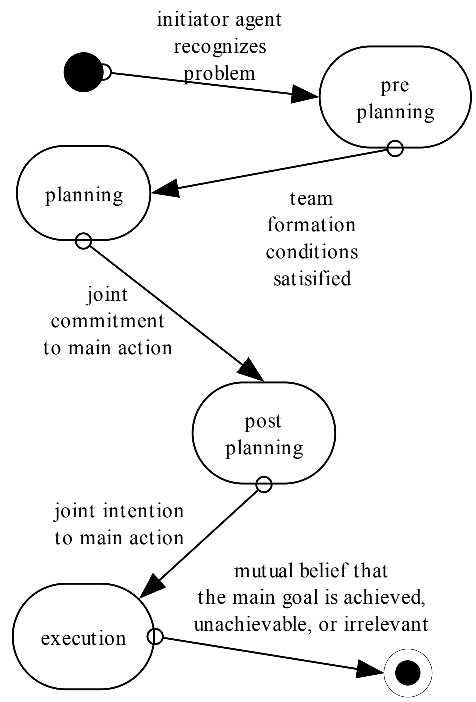

Figure 1: State chart of CPS process

A teamwork conversational pattern consists of (i) a set of valid team states, (ii) a set of preconditions, (iii) a partial order of exchanged teamwork interaction messages and fired deliberation rules qualified by agent roles, and (iv) a set of post conditions. The set of valid team states are those at which the conversational pattern are applicable.

The preconditions states the set of constraints that should be matched in order for the pattern to be matched. The messages exchanged outlines agent interactions. The fired deliberation rules represents decisions taken by agent as a result its reasoning about teamwork. Pattern pre (post) conditions states the constraints that indicates the start (end) of a given scenario. The interaction situation grows as the set of interaction messages and deliberation rules are being matched. Figure 2 illustrates a partial extraction of the *form_team* conversational pattern, outlining pattern elements.

The teamwork conversation policy guides agents teamwork behavior during a number of interaction situations. A conversation policy is defined as [9]:

"...a collection of rules and interaction specifications that guide a particular path or trajectory in a conversation space..."

The proposed teamwork conversation policy is comprised of (i) a set of domain and problem specific rules to be defined by the developer of multi-agent system, (ii) a set of teamwork rules explicitly defined by the proposed framework state model, and (iii) a set of teamwork rules defined by the underlying models as described in [8] and [7].

Form Team Conversational Pattern

Valid Team States:
Pre-Planning

Pre-Conditions:
the team formation facilitator agent has a designated goal to achieve through team action.

Step 1: Reasoning Performed by the Team Formation Facilitator Agent
belief that a candidate team is able to achieve designated goal.

as a result, attempt to solicit assistance in order to form team to achieve designated goal.

Step 2: Messages Sent by the Team Formation Facilitator Agent
announce to candidate team agents its attempt to solicit assistance in order to achieve designated goal.

Step 3: Reasoning Performed by Other Candidate Agents
maintain belief about team formation facilitator announced attempt.

Assumption
negotiation might be performed at this stage. for example, in the trade domain, negotiations of price, payment terms and delivery method takes place.

Step 4: Reasoning Performed by All Candidate Agents
individually reason that candidate team is able to achieve designated goal.

Figure 2: A partial extraction of the *form_team* conversational pattern

FRAMEWORK IMPLEMENTATION

In this section, we discuss elements of the proposed framework implementation layer based on the conceptual model defined above. The proposed framework implementation is one of a set of possible framework implementations, *varying in their component definitions and enabling the creation of implementations that match the configurations of open environment*, which define details of messaging, ontology, communication and interaction mechanisms. A possible framework implementation would consist of the following components: Behavior Representation Language (BRL) that is used to exchange information about agent behavior, Agent Communication Language (ACL) that allows agent to express and query other agents' mental state, and Teamwork Message Format for representing team messages in XML along with a message encoding and decoding facility.

The Behavior Representation Language (BRL)

The behavior representation language (BRL) enables team agents to express their own and understand other agents' teamwork behaviors. The language defines several types of mental elements namely: Belief, Goal, Commitment, Action, and Intention. For example, an agent might be committed to achieve a designated goal or to execute a designated action. An intention might be associated with a given action. An agent might adopt a belief about the existence of a designated mental element. An agent might have a goal to achieve relative to some condition defined through the existence of a designated mental element.

BRL acts as ontology for the framework implementation in the sense that it is a specification of the objects, concepts, classes, functions, and relationship defining collaborative mental behavior fore an agent. The ontology can be shared and reused by many agents speaking common vocabulary of BRL within the domain of collaborative mental behavior. A partial excerpt of BRL language object model is depicted in figures 3(a) and 3(b), showing generalization and associations relationships between some BRL language elements.

BRL is defined in order to represent agent knowledge-level behavior. For example, committing to an action, or dropping a belief is characterized as a knowledge-level behavior. BRL language defines a set of clause types. Our approach to represent language constructs is through the encoding of language clauses into a carrier structure. In the proposed framework implementation, the carrier structure is XML. Every language clause has its own mapping into XML node.

BRL enables the extension of semantic attributes associated with BRL language clauses using BRL extension mechanism. For example, an agent can express its commitment to a designated action, adding a semantic attribute that represents commitment expiry after a given time interval. As a result, the different varieties of attributes cannot be enumerated at the time of framework implementation. For this reason, the implementation should provide the flexibility that would allow multi-agents system implementers to define their own semantics. This flexibility is provided using XML as the carrier format for BRL and Message Encoding/Decoding facility.

The Agent Communication Language (ACL)

The agent communication language is used by an agent to express and inquire mental states of other team agents. ACL supports the inform and inquire speech acts, and the inform and the inquire interaction mechanisms. An agent uses the inform mechanism to inform a set of agents about the status of its mental behavior by sending a message clause with the truth value attribute set to either true or false. An agent receiving such a message is expected to update its mental state accordingly. On the other hand, an agent uses the inquire mechanism to inquire the existence of a specific mental element within some other agent's mental state. An agent receiving such an inquiry is expected to reply back declaring the status of existence of queried mental element as part of its own mental state.

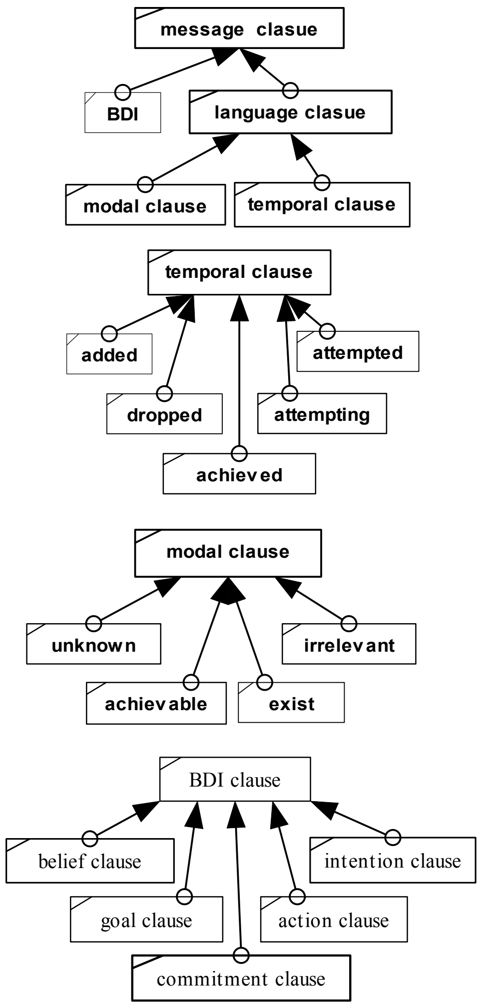

Figure 3(a): A class diagram showing generalization relationships between knowledge elements

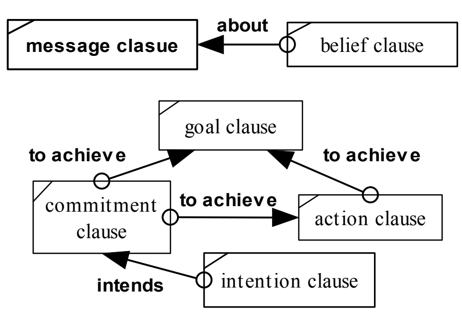

Figure 3(b): Possible association relationships between BRL knowledge elements

The Teamwork Message Format

The teamwork message format encodes an agent's message identification and delivery information, ACL's speech act used, BRL language clauses describing agent mental behavior. An application class library is used to encode and decode teamwork messages. Each language clause maps to an XML node and each of the clause's semantic elements, pre-defined or user defined, maps to an XML attribute. Figure 4 shows a sample teamwork message, classified into the following elements:

- a number of header nodes encoding message team and team ids, and framework message version
- delivery information node defining the sender, recipient and reply to lists
- message content node defining main message clause along with associated Belief-Desire-Intention (BDI) clauses.


```
<team_message team_id = "trade_team"
message_id = "attempt_to_solicit_assistance">
  <meta_info content_type =
"framework-implementation-horn-clause"/>
<delivery_info sender_id="buyer"><recipient_list> <recipient
agent_id = "merchant"/> <recipient agent_id = "delivery"/>
</recipient_list>  <reply_to_list><recipient agent_id =
"merchant"/> <recipient agent_id = "delivery"/><recipient
agent_id = "buyer"/> </reply_to_list> </delivery_info>
<message_content> <message_clause truth_value = "true"
bdi_clause_id = "SolicitAssistance" type = "added" id =
"SolicitAssistance"/> <bdi_clause entity_id = "buyer" type =
"goal" id = "SolicitAssistance_goal"/> <bdi_clause action_id =
"SolicitAssistance_action" entity_id = "buyer" commitment_id
= "SolicitAssistance_commitment" type = "intention" id =
"SolicitAssistance_intention"/> <bdi_clause entity_id =
"buyer" type = "action" id = "SolicitAssistance_action"/>
<bdi_clause entity_id = "buyer" type = "attempt" intention_id
= "SolicitAssistance_intention" id = "SolicitAssis-
tance_attempt"/> <bdi_clause entity_id = "buyer" goal_id =
"SolicitAssistance_goal" type = "commitment" id =
"SolicitAssistance_commitment"/> </message_content>
</team_message>
```


Figure 4: A sample team message

THE E-COMMERCE CASE STUDY

In this case study, we develop a model of collaborative agents that executes an e-commerce purchase. The developed trade teamwork model builds on open multi-agent collaboration framework proposed. The case study illustrates the impact of using the proposed open collaboration framework on the development of trade agents' reasoning model.

We start by identifying teamwork aspects in the purchase process and then, we define knowledge level model for reasoning about collaboration within a team of agents conducting an e-commerce purchase transaction, and finally, we illustrate the use of an iterative development approach for defining multi-agent collaborative behavior.

Teamwork Aspects in the Purchase Process

The purchase process can be conceived as a process of co-operative problem solving (CPS) in which team agents succeed or fail together. The actions of a purchase transaction can be classified into (i) individual agent actions and (ii) joint agent group actions. Action relationships can be depicted as a directed graph, with its actions representing graph nodes, and its actions execution dependencies representing graph edges.

Trade team members reason individually about collaboration act in a manner that benefits the overall team. For example, if the merchant agent has an individual belief that the goal of performing a purchase cannot be fulfilled, it shares this belief with the buyer and delivery agents in an attempt to reach a mutual belief about that main goal. As a result, the trade team shall be able to update its execution plan in order to gracefully roll back the purchase transaction. The modeling of e-commerce purchase as a cooperative problem solving is defined in table 1.

Table 1: Modeling the e-commerce purchase as a process of cooperating problem solving

CPS Phase	Purchase Phase	Description
Recognition	Decision to buy merchandise	The buyer agent recognize the potential for performing purchase
Pre-team	Trade negotiation phase	Buyer agent negotiates with a group of merchants agents to select the best offer, and then, construct a trade team in order to execute the e-Purchase transaction
Planning		Buyer and merchant and delivery agents agree on all purchase attributes
Execution	Value exchange phase	Trade team members execute purchase transaction steps in a timely coordinated manner
	Trade settlement phase	

A Knowledge Level Model for Reasoning about Collaboration

Next, we define a knowledge level model for reasoning about collaboration within a team of trade agents. Belief-Desire-Intention architecture (BDI) is assumed for trade agent. The defined knowledge model consists of a group of mental elements representing an agent's mental state. Mental elements are classified into as a set of beliefs, goals, commitments and intentions that team agents adopt as a result of its own reasoning about the mental status of itself and the mental status of other team agents. For each knowledge element, the model defines (i) the owner agent, (ii) mental element type (individual or joint), and (iii) add and drop triggers. The add trigger defines conditions that result in adoption of the knowledge element. The drop trigger defines conditions that result in dropping off that knowledge element.

An agent relies on proposed framework to reason about teamwork state using teamwork state model and to reason about interaction using teamwork conversational pattern. In addition, a trade agent relies on proposed framework implementation to express and query mental states of other team agents through the use of agent communication language (ACL), behavior representation language (BRL), and teamwork message format..

The Use of an Iterative Development Approach for Defining Multi-agent Collaborative Behavior

Using an iterative development approach, the trade agent team designer starts by developing a prototype of trade agents' teamwork behavior along with input scenarios used to test the prototype. Teamwork scenarios represent typical interaction situations that an individual trade team agent is required to reason about and act upon. The prototype is verified against every input scenarios defined, and the overall teamwork behavior is analyzed at each input scenario. Returned feedback might demand a change to the prototype, which should be verified against input scenarios and then changed as required until teamwork behavior required is reached.

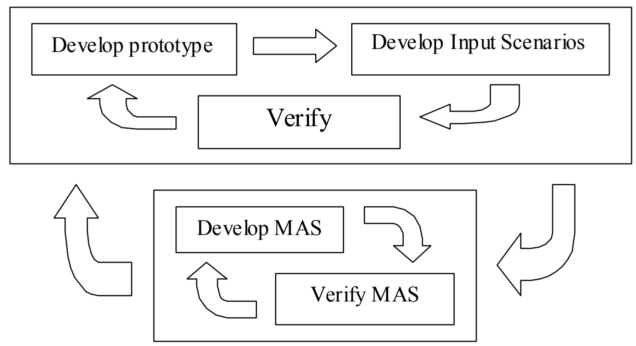

Figure 5: An iterative approach for the rapid prototyping of MAS collaborative behavior

As the teamwork behavior defined in the prototype matures, the teamwork behavior prototype along with developed input scenarios are treated as a specification of teamwork behavior that can be used as an acceptance criteria during the development and testing of actual trade agents. Figure 5 illustrates the proposed approach.

As an application of the proposed approach, a prototype of a trade team were developed, where the behavior of each trade agent is simulated to imitate the execution the trade purchase. The prototype covers the following team interaction scenarios:

1. Form a trade team by having the buyer soliciting assistance from the merchant and the delivery agents
2. Build an action commitment hierarchy, and
3. Simulate trade action execution, through building a hierarchy of intentions

At each of these cooperative problem solving phases, each simulated trade agents maintain beliefs about other agents' commitments, intentions, and action execution status. The prototype's collaboration strategy is defined by the following teamwork rules:

- If an agent drops a commitment to execute an action, then the agent on which the dropped action is dependent should drop its commitment
- If the initial purchase sub-action is dropped, then the agent should drop joint commitment to main purchase action

- If an agent's intention to the joint action is dropped, then other agents should drop their intentions to that action
- If an agent drops its intention to execute an action, then the trade behavior would freeze
- If an action attempt fails, then the agent would drop its intention to execute that action

The developed prototype was tested against a varied number of interaction situations. Test results shows that the developed teamwork agents prototypes were able to clearly reason about teamwork states and interaction and were able to act in a way that benefits the whole team members.

The Findings

The case study findings can be summarized as follows:

- Agent interaction and communication are crucial for maintaining a shared and consistent view of the trade problem. Agents were required to express their mental states in a clear and consistent manner understandable by other agents.
- Interaction enabled agents to reason about coordination and cooperation.
- A common view of the goals, actions, commitments, and intentions, were required since this view helped agents to make their own decision regarding teamwork activities and state.
- The use of framework conversational model helped agents contribute to interaction scenarios required to develop and maintain team's mutual understanding and team's shared view of the trade. In addition, it enabled agents to consistently reason about team states.
- The framework implementation enabled agents to express collaborative mental behavior, using a set of agent interaction mechanisms, and transmitted using a message interchange format.
- Reviewing the results of executing input scenarios against the developed prototype enables the verification of team and individual collaborative behavior.

SUMMARY AND CONCLUSION

Current MAS environments that support multi-agent collaboration lacks the characteristics of an open environment. On the other hand, current open multi-agent environments lack the existence of a general collaboration framework required to supports the development and maintenance of shared team mental state among team agents. Existing multi-agent standards, development tools, and environments support the development of MAS through providing support for communication, planning, coordination, cooperation, task allocation and task execution activities.

The proposed framework does not assume the existence of a fixed set of interaction protocols. In addition, the proposed framework is transparent to existing tools since there is no overlap in functionality between the proposed framework and these tools.

The framework design maintains a clear separation between the framework specification and possible framework implementations, enabling the creation of implementations that match the configurations of open environment.

The proposed framework implementation provided a behavior representation language that supports modal and temporal constructs, enabling agents to reason about the possibility and time. In addition, it provided an agent communication language that allows agent to express and query other agents' mental state. The proposed framework implementation also assumes a layered conversational model, it also defines a teamwork message format for representing team messages in XML along with a message encoding and decoding application class library.

The use of collaborative agents model for representing trade processes would enhance the flexibility and dynamism for such heterogeneous systems of e-commerce, for example, in the trade domain, if the payment fails due to the fact that the credit card has expired, an agent might try to use another credit card in case one exists, and in case it would also fail, the trade agent might re-negotiate the payment method before announcing the failure of payment action.

The trade purchase problem can be modeled as a number of trading entities acting together to select, negotiate and execute trade transactions, facilitated by the use of the framework's specification and implementation.

The iterative proposed approach for prototyping trade team agents' collaborative behavior enables the rapid development and maintenance of the specification of MAS collaborative behavior. Test results shows that the developed teamwork agents prototypes were able to clearly reason about teamwork states and interaction and were able to act in a way that benefits the whole team members.

REFERENCES

[1] A. Lomuscio, M. Wooldridge and N. Jennings, "A Classification Scheme for Negotiation in Electronic Commerce" in Agent-Mediated Electronic Commerce: A European AgentLink Perspective (eds. F. Dignum and C. Sierra), Springer Verlag, 19-33, 2001.

[2] Q. Chen, M. Hsu, U. Dayal and M. Griss, "Multi-Agent Cooperation, Dynamic Workflow and XML for E-Commerce Automation", Proceedings of the fourth international conference on Autonomous agents, June 2000.

[3] G. Weiss, Editor, "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence", MIT Press, 1999 .

[4] M. Dastani, J. Hulstijn and L. Torre, "Negotiation Protocols and Dialogue Games", Proceedings of the fifth international conference on Autonomous agents, July 2001.

[5] N. Jennings, E. Mamdani, I. Laresgoiti, J. Perez and J. Corera, "GRATE: A General Framework for Coopera-

tive Problem Solving", IEE-BCS Journal of Intelligent Systems Engineering, 1 (2), 102-114, 1992.

[6] M. Tambe, "Towards Flexible Teamwork", Journal of Artificial Intelligence Research, 1997.

[7] P. Cohen and H. Levesque, "Teamwork", Nous 25(4), Special Issue on Cognitive Science and Artificial Intelligence, pp. 487-512, 1991.

[8] M. Wooldridge and N. Jennings, "Cooperative Problem Solving" Journal of Logic and Computation 9 (4) 563-592, 1999.

[9] M. Nowostawski, M. Purvis and S. Cranefield, "Modeling and Visualizing Agent Conversations", Proceedings of the fifth international conference on Autonomous agents, July 2001.

[10] M. Wooldridge and N. Jennings, "Intelligent Agents: Theory and Practice", The Knowledge Engineering Review, 10 (2), pp. 115-152, 1995.