# Cooperative Dialogues with the Support of Autonomous Agents

*Geof Staniford**, *Trevor J.M. Bench-Capon, and Paul E.S. Dunne*

*University of Liverpool – Department of Computer Science*

**Abstract**

*An architectural specification of independent communicating agents is described. Such agents are designed for use in an environment which is most suitably represented as a graph or network and an example of one such environment is given based on the structured document graph paradigm. The specifications extend agent architectures in a natural way so that the complexities found in distributed systems may be addressed. Such an approach has a number of advantages; it provides a general representation paradigm that can be used as a basis in a wide range of intelligent distributed systems and is easily customised to cope with problems requiring graphs with differing structures and with different node and edge properties and attributes. A suitable platform to build agents and environments and an example of cooperative communication using dialogue analysis is presented.*

## 1. Introduction

*"I believe that Calcutta exists, though I do not perceive it, and that it would still exist if every percipient inhabitant were suddenly to leave the place, or to be struck dead. But when I analyse the belief, all I find in it is, that were these events to take place, the Permanent Possibility of Sensation which I call Calcutta would still remain; that if I were suddenly transported to the banks of the Hoogly, I should still have the sensations which, if now present, would lead me to affirm that Calcutta exists here and now."* *J. S. Mill*

At present, the design and implementation of *multi-agent systems* (MAS) is a very active and fluid area of *distributed artificial intelligence* (DAI) research. One major approach is to view and model the human mind as a set of cooperating agents, Minsky [17]; another approach is to use communicating agents to design more intelligent distributed systems, Bond & Gasser [4]. In this paper we concentrate on the second paradigm.

The process of collaboration through structured dialogue is a common and valuable method of reaching an agreed position. One problem with this approach, however, is that the conclusions reached by a dialogue and the arguments by which they are arrived at may have to be consolidated into a single document reporting on the discussion. If this task is undertaken by one of the participants it is possible that the final report may fail to truly reflect the collaborative process. Ideally an external, independent observer acting as a *rapporteur* should be employed to carry out this activity.

In this paper we present an architectural specification of independent communicating agents which provides the basis upon which we build a *Rapporteur Agent*. The system of autonomous agents is designed around the concept of a 4-layer model of agent communication and cooperation—Staniford & Dunne [22], Staniford [21]—and is intended to enable the creation of documents represented in machine form using any variant of the graph-theoretic paradigm. The agents employ knowledge engineering techniques to reason about their domain and to control the asynchronous interaction of several authors. We describe a platform upon which this architecture has been implemented and give an example *argument* to show how these elements may be combined in producing documents that summarise the content of a dialectical discussion. A simple model of dialogue structure and content, which capture certain elements of general discussion, is employed for this purpose. The system is based on the concept of formal dialogue games, McKenzie [13], Bench-Capon *et al* [1], Bench-Capon *et al* [3], and employs graph grammars as a mechanism for transforming the dialogue form into a document; this approach is based upon the *Rapporteur System* introduced by Bench-Capon *et al* [2] and follows the graph modification system introduced in that work very closely, but implements the concepts using agents designed in accordance with the architectural scheme outlined in this paper.

In designing these architectures and systems the

* This research was supported by SERC research grant No. 90309983

authors intend to provide a platform for the study of cooperation between autonomous communicating agents This approach is discussed and future directions are outlined in the final section.

## 2. Loquacious-agents and their environment

Genesereth & Nilsson [11] define four types of agent, *tropistic*, *hysteretic*, *knowledge level* and *stepped knowledge level*, in order of increasing sophistication and complexity. These agents are designed to exist in an environment in which they have the sole occupancy. There is no need for them to be aware of nor communicate with others of their kind. Staniford and Dunne have extended this architecture to enable agents to coexist with, communicate with and cooperate with other agents. For the sake of brevity we summarise the definitions here—note, that we only describe any given set once, and this description serves for all definitions in which the set occurs—the interested reader is referred to Staniford & Dunne [22] and Staniford [21] for a full account. So that an agent may receive external stimuli we require two functions:

*Definition 1:*

$$see : R \rightarrow S$$

The set $R$ is the set of states that characterises the agents world. In order to characterise the agents sensory capabilities we partition the set $R$ of external states into a set $S$ of disjoint subsets such that the agent is able to distinguish states in different partitions but is unable to distinguish states in the same partition •

*Definition 2:*

$$hear : T \rightarrow Q$$

The set $T$ is a set of words that characterise communications that the agent may receive from other agents, and similar to the foregoing in order to further characterise the agents sensory capabilities we partition the set $T$ of words into a set $Q$ of disjoint subsets such that the agent is able to distinguish words in different partitions but is unable to distinguish words in the same partition •

These are sensory functions and are used to characterise the way in which an agent perceives stimuli external to itself. It will sometimes be necessary to define agents that carry out an action purely as a result of the influence of only one of the two external stimuli, consequently it seems reasonable to keep the two functions separate. Following Milner [15,16] *hear* as an act of communication is considered to be an indivisible event, taking place the instant that a message is available from another agent. Similarly we consider that any local attributes of a state in an environment that an agent can "see" will be available for perception by that agent as soon as some state change within the environment has occurred.

Turning our attention to the manner in which the state of an environment is changed, we assume that agents make only local changes to an environment. In order that an agent may make changes we define two more functions:

*Definition 3:*

$$action : S \times Q \rightarrow A$$

is a discriminatory function which maps each disjoint subset of states and inputs onto a particular action or set of actions to provide an agent with the ability to choose which action to perform according to its perceived stimuli •

*Definition 4:*

$$do : A \times R \times T \rightarrow R \times W$$

is an executory function which maps an action, state and input onto the new state and an output; providing the agent with the ability to change the local state of its environment and to communicate a message to another agent. The set $W$ is a set of words that characterise the communications that an agent may send to another agent. We assume that the agent can distinguish all its responses so there is no need to partition $W$ •

Unlike *see* and *hear* we encapsulate acting and communicating in one function because although there will be occasions when we wish to communicate without changing a state, the converse is not the case. We do not allow an act which changes the state of the environment to take place without there also being a corresponding act of communication; hence the use of the word loquacious in our name for these agents. We wish to indicate that these two operations are closely bound together in order that we may simplify the coordination of the knowledge—between autonomous agents—that environmental state changes have taken place. The ability to i communicate without state changes is in fact crucial to the notions of cooperation that we will develop later. Although we do not explicitly map $W$ into a set of disjoint subsets, implicitly outputs fall into one of three categories; successful completion of a state change, failure to complete a state change and communications regarding cooperation. In order to avoid ambiguity we require that these categories do form disjoint subsets within $W$ and in fact that there is an implicit bijection between the success and failure messages.

For our purposes then, acts of observation both visual and oral must take place in the internal state of an agent, between changes in state of the external environment. In general there is no simple linear correlation between the internal state changes within an agent and the external state changes of the environment. This presents problems for analysis and design, but we contend that these problems are more amenable to solution with the use of communicating agents than with simple non-communicating agents.

A *tropism* is the tendency of biological life forms to react in response to an external stimulus and our simplest agent - *tropistic* - is modelled with this metaphor in mind.

*Definition 5:* A *loquacious tropistic agent* in an environment is a 10-tuple of the form

$$< R, S, T, Q, W, A, \mathit{see}, \mathit{do}, \mathit{hear}, \mathit{action} >$$

where the sets and functions are all as previously described. •

Hysteresis carries with it the notion of a simple form of memory, of a previous state having an effect on the next state; this notion is used to increase the complexity of an agent.

*Definition 6:* A *loquacious hysteretic agent* in an environment is a 12-tuple of the form

$$< I, R, S, T, Q, W, A, \mathit{see}, \mathit{do}, \mathit{hear}, \mathit{internal}, \mathit{action} >$$

where the sets and functions are all as previously described except that the set $I$ is an arbitrary set of internal states and we assume that the agent can distinguish between all its internal states so that there is no need to partition $I$. The function $\mathit{internal} : I \times S \times Q \rightarrow I$ maps an internal state and both types of observation into the next internal state. Finally, the function $\mathit{action} : I \times S \times Q \rightarrow A$ maps each internal state, external state partition and input partition into the action that the agent is to perform whenever it finds itself in a particular combination of internal states, inputs and external states. •

Moving on from simple facts we allow that it will be necessary to consider relations between facts in influencing the actions of an agent in its environment.

*Definition 7:* A *loquacious knowledge level agent* in an environment is a 12-tuple of the form

$$< D, S, T, A, R, Q, W, \mathit{see}, \mathit{do}, \mathit{hear}, \mathit{database}, \mathit{action} >$$

where the sets and functions are all as previously described except that the set $D$ is an arbitrary set of predicate calculus databases, The function $\mathit{database} : D \times T \times Q \rightarrow D$ maps a database and both types of observation into the new internal database. Finally, the function $\mathit{action} : D \times T \times Q \rightarrow A$ maps each internal database, external state partition and input partition into the action that the agent is to perform whenever it finds itself with a particular combination of internal databases, inputs and external states. •

When a sequence of different action patterns is proposed, interactions occur not only between feedback and the central control of the actions themselves, but also between feedback and the central control of groups of actions Hinde [12].

*Definition 8:* A *loquacious stepped knowledge level agent* in an environment is a 12-tuple of the form

$$< D, S, T, A, R, Q, W, \mathit{see}, \mathit{do}, \mathit{hear}, \mathit{database}, \mathit{action} >$$

where the sets and functions are all as previously described except that the the function $\mathit{database} : D \times N \times T \times Q \rightarrow D$ maps a database, cycle number and both types of observation into the new internal database. Finally, the function $\mathit{action} : D \times N \times T \times Q \rightarrow A$ maps each internal database, cycle number, external state partition and input partition into the action that the agent is to perform whenever it finds itself with a particular combination of internal databases, inputs and external states. •

Notice that the only difference between a *stepped knowledge level agent and* a *knowledge level agent* is the dependence of the database and action functions on the agents cycle number.

In order that agents may reason about their own sequences of actions we introduce the idea of *deliberation*. Deliberation allows us to design some sequences of actions based on *classical decision theory,* McCleery [14] or on the notion of *satisficing,* Simon [20] or on the principle of *resolution* Robinson [19].

*Definition 9:* A *loquacious deliberate stepped knowledge level agent* in an environment is a 12-tuple of the form

$< D, S, T, A, R, Q, W, see, do, hear, database, action >$

where the sets and functions are defined identically to those in definition 4. The difference in defining agents in this class arises from the use of an automated inference method like *resolution* when defining the operation of the *action* function - the authors are currently actively investigating the use of *Argument Schemas*, Toulmin [24], in deriving a sentence that indicates the required action on each cycle. An agent of this sort is *deliberate* in that it deliberates on every cycle about which external action to perform. •

There are two primary mechanisms that may be used to encapsulate autonomous agents into higher level structures; the vocabulary of the agent may be used to prescribe which other agents are allowed to meaningfully communicate with it and we can use the computational ability of an agent to implement cooperation strategies that enforce a hierarchy of control.

Agents may be grouped together in systems etc. of cooperating equals by means of intra-layer cooperation strategies. We do not view such a grouping of architecturally similar agents cooperating upon a common task as a complex agent, such a grouping is a set of autonomous communicating agents, that we might describe with collective nouns such as team, system, sub-system, society, etc. Precisely which form of strategies to choose is currently an open question which will form a major part of our forthcoming research.

A discussion report is a simple example of the more general document models described in Dunne & Staniford [9, 10].

*Definition 10:* A *report graph*, $G_r(V_r, E_r)$, is a directed acyclic graph. The vertices in $V_r$ denote *objects* in the report and the edges in $E_r$ depict logical connections between the objects. Each object has an associated *object type* which consists of two parts: a *data type* which specifies the domain of possible data values for the object (e.g. word, phrase, sentence etc. ); and an *attribute type* which indicates the domain of possible properties that the object may posses (e.g. font, size etc.). Objects may also be labelled •

*Definition 11:* A *report specification* consists of a pair $RS = (C, Init)$, where $C$ is a finite set of *constraints*,

$$C = \{ C_1, C_2, \ldots, C_k \}$$

where each $C_i$ is a computable predicate on report graphs. *Init* is a set of *initial* report graphs. Given a report specification $RS$ and a report graph $G_r$, $G_r$ is said to meet the specification $RS$ if and only if $G_r \in Init$ or $C_i(G_r)$ is true for each constraint $C_i$ •

These notions, we contend, when taken with the agent architectures described allow the design of systems that fit in very well with the ideas contained in the top down design philosophy first mooted by Dijkstra [8] but enable the extension of those ideas to include the notion of cooperating equals in a system.

## 3. A platform for prototyping agents

An attractive proposition for workers within the logic programming paradigm intending to build multi-agent systems would be the enhancement of an existing language that is already in widespread use for programming knowledge intensive applications. We consider, very briefly, one such approach used by the first author in practical development work.

IC-Prolog II (ICP) Cosmadopoulos & Chu [6] has had a number of features added over first generation Prologs in order that workers may program distributed systems at a high level. In particular we note that ICP is multi-threaded. Each thread is a process with its own execution and variable storage area and there are well defined primitives by which threads may communicate with each other. Threads in one main ICP process do however share the one prolog database, which does mean that considerable discipline is required when programming. Processes may communicate by one of three methods, pipes, TCPIP primitives and mailboxes. Pipes have been specifically designed as a uni-directional channel of communication between threads running under the same ICP process and are supported by a rich set of primitives. Separate ICP processes running on the same or different machines may communicate via ICPs' built in TCPIP primitives; those who wish to program at this level of abstraction should refer to Cosmadopoulos & Chu [6] and Stevens [23].

Mailboxes also provide a means of process/process communication that has been designed both as a means of enabling workers unfamiliar with low level communications programming to use very high level constructs in building experimental communications prototypes and as a transparent method of minimising network resource consumption, Chu [5]. Each individual TCPIP connection—used transparently by the mailbox program—supports up to sixty four virtual channels and each virtual channel appears as a FIFO queue at the mailbox to communicating processes. A full set of primitives is provided for use with mailboxes.

## 4. An example using Rapporteur

It is not possible in a paper of this length to discuss the properties of all four types of agent architectures; with this constraint in mind we leave the simpler agents and look at the most complex. The author, using ICP as a platform and the architectures given previously, has developed a number of communicating agents that are currently under testing and further development, in this section we discuss informally the functionality of two of the agents that have been implemented.

One highly fruitful way in which people collaborate when working on a subject is through dialectical discussion: one person tries to establish a point while his colleague, either because of genuine scepticism or because he is playing devil's advocate for the purposes of the discussion, attempts to rebut the point. Such a discussion will help to structure the argument, clarify the position, and anticipate objections which require either additional exposition or refutation, or else which require the original position to be modified or withdrawn. When the aim is to produce a documented record, the discussion must be be followed by a writing process in which the resulting developed argument is committed into an electronic version of the document. In our system when such dialectical discussion occurs it is recorded by an *autonomous agent* acting as a *rapporteur* whose responsibility is to synthesise what may be a rambling discussion into a coherent document setting out the thrust of the debate. Our *rapporteur* agent is intended to support two or more participants—although only two are shown in Figure 1, for clarity—collaborating through dialectic, and to produce a report of their discussion. This is achieved by conducting the discourse according to the rules of a dialogue game, and producing the report using a formal document specification, see Bench-Capon *et -al* [2].

Games can be viewed as *autonomous normative systems,* Raz [18]. Systems which contain a set of *structural rules*, a set of *mandatory norms*, and a set of *values*. The successful transfer of such systems from the board etc. to computer programs is not open to question. *Rapporteur* agents are designed to allow collaborative participation in a constructive dialogue game according to well defined principles given as below. One set of participants must adopt the role of proposer; making an initial assertion and then taking turns to provide arguments in support of that assertion. The other participant(s) adopt an opposition role in which challenges and objections to the proposers assertion and supporting premises are put forward. The *rapporteur* agent allows counter objections and makes provision for both sides to modify earlier arguments. Either side can win the argument; in the case of the opposition being successful, the original assertion must either be negated or withdrawn.

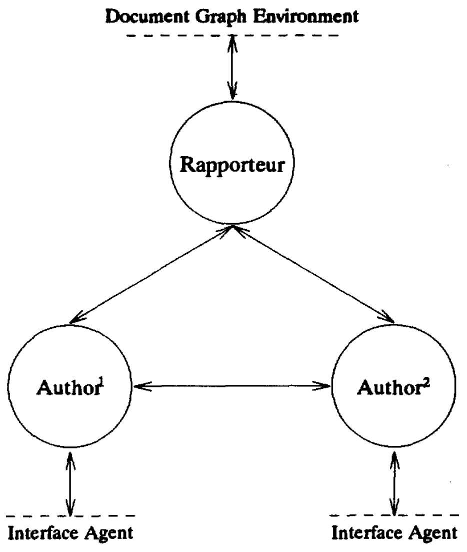

Figure 1: A communicating sub-system of Ls-Agents

Both sides take turn and turn about in presenting their respective cases, although one member of a side may take several consecutive turns for that side in order to present a particular line of thought. Game play takes place in a structured way which reflects the different roles that the two sides bring to the dialogue. The proposers are required to present an assertion; are allowed to modify that assertion, provide supporting premises and modify those premises, refute objections from the opposition and require the opposition to continue objections and challenges. In their turn the opposition are allowed to challenge the assertion or premises, object to premises and modify those objections and require the proposers to continue the assertion, premises and refutations. Either side may accept defeat, the opposition by accepting it has no valid challenge or objection, the proposers by accepting they have no valid refutation of the claim.

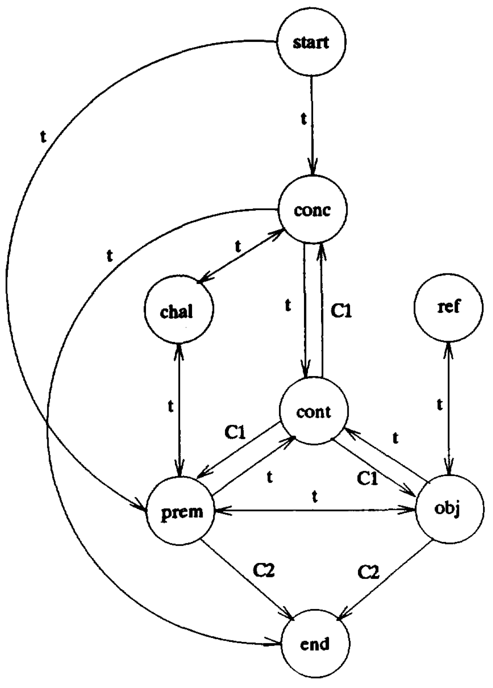

Figure 2: Rapporteur's Dialogue Control Graph

The *Rapporteur* agent oversees the game and will only allow legal moves to be made, imposing the *structural rules* and *mandatory norms* upon the participants; the *continuity norms* are considered to be implicit within the situation that the dialogue game will be played. The *values*, of trying to win or not to lose are the normal *opearative first order reasons* for action and these are totally in the domain of the participants. *Rapporteur* has several general dialogue graph models one of which is presented in Figure 2., sequences of legal moves can readily be worked out from this graph. Edges in the graph may only be traversed if the condition attached to the edge can be satisfied; informally C1 states—given *last node*, *current node*, and *next node*—that the the next legal edge is the one where label( *next node* ) = label( *last node* ) and C2 requires that label( conc ) is in the history list. Edges marked t may be traversed at any time in the direction of the arrow. Not evident from the graph, because omitted for the sake of clarity, is *rapporteur's* ability to control some of the finer grained complexities of human arguments. For example the agent knows about many *conditionals* that we commonly use eg. given a *conditional* if *P then Q* were *Q* is some premise in a dialogue, *rapportuer* would not allow the argument to proceed until *P* had been explicitly asserted.

Report graphs are *abstract representations* of report structure: the form that is manipulated during the generation of the report from the raw dialogue graph. Our objective is to have this abstract representation closely matching the dialogue participant's conceptual representation of an accurately summarised logical discussion.

We thus have two types of graphs, one — a directed cyclic graph, representing the realised dialogue space, and one — a set of directed acyclic graphs, representing legal models of a report of a dialectic discussion; both types of graph containing single sources and sinks. The main task facing the *rapporteur agent* is to transform the former into the latter. This will, for example, include moves to enable the digressions common in dialectic, such as when a person puts forward a definition which is found by a challenge to be inadequate, and which is consequently modified, to be elided so that the report will show only the final form of the definition, and the debate which led to the modification will be included in the justification of that definition.

To this end, during the course of a dialogue, the *rapporteur* agent explicitly builds a set of nodes $V_d$, whilst implicitly following a set of edges $E_d$ of the dialogue graph $G_d(V_d, E_d)$. To achieve a mapping between the two graphs the agent adopts the strategy of partitioning $V_d$ such that a set of candidate nodes, $SP_d$ say, includes all nodes in $V_d$ that lie on the shortest path through $E_d$, and then discards nodes that lie in $V_d - SP_d$. Depending on which side won the game, the agent partitions $SP_d$ to produce $V_r$. The agent has no notion of the semantics of any argument; it does however have specifications for a number of syntactically correct propositional arguments. Applying these specifications the agent produces a mapping from $G_d(V_d, E_d)$ to $G_r(V_r, E_r)$, a legal report graph. At the present time the allowed mappings do not form conflict sets but in future it is intended that the agent should have an increase in the number of specifications and conflict resolution will be necessary.

## 5. Example dialogue

As an example of the kind of dialogue which we can analyse in this way, consider the following argument concerning the sterilisation of surgical implements. We will suppose that a particular kind of organism has been identified, and the the junior member of the pair proposes:

Boiling water kills these organisms immediately.

No. *This objection is made because the senior partner knows that these organisms survive temperatures of up to 103$^O$ C.*.

Why not? *The objection was not expected, so explanation is sought*.

The organisms die at 103$^O$C. *The reply given is the most important reason for the objection, i.e the data.*

And? *The junior partner is still not convinced*

Water boils at 100$^O$ C. *The other premise - not given first because $P^1$ was expected to know this.*

But the water was boiled in Portsmouth. *$P^1$ detects the reasoning flaw and sets up a rebuttal of the objection.*

So? *The senior partner knows nothing to make this new fact relevant*.

Portsmouth water boils at 104$^O$ C. *This is the basis for the rebuttal*

Why? *This contradicts the senior partner's default knowledge, so justification is required*

It contains impurities *This is the other item of data required to establish the rebuttal*

So? *$P^2$ now seeks the rule which led to the inference*

Impure water does not boil at 100$^O$ C.

Notice in particular how the basis for the rebuttal of the original claim has itself, when challenged by $P^2$, become the claim of a subsidiary argument; an argument that is shown to be false and consequently would not appear in the argument report. The example also illustrates how a junior, with less background knowledge, may sometimes be in a position to contribute significant specialised information in the course of a dialogue; $P^1$ is not an expert on the survival of bacteria, but does happen to know a curious 'fact' about water in Portsmouth.

This dialogue would generate the the following report.

Portsmouth water boils at 104$^O$ C and the water was boiled in Portsmouth and the organisms die at 103$^O$C therefore boiling water kills these organisms immediately.

## 6. Discussion and conclusion

Architectural models for autonomous|semi-autonomous communicating agents have been described. The models provide agents with differing degrees of complexity that may be used for different purposes in larger overall systems. The environments upon which the agents may act was specified as directed acyclic graphs which were viewed as being in a dynamic state by the use of graph grammars. Taken together, the architecture and the environment provide a rich, specifiable resource with which to build, predict and study the behaviour of autonomous communicating agents.

An example was presented—to illustrate some of the possible functionality of the most complex of the agent types—that showed that an agent such as *rapporteur* provides a way of enforcing a general syntactic structure to a dialogue represented by a graph. This structure is sufficiently flexible to allow the participants to conduct their dialogue using deduction, induction or indeed abduction as the mode of reasoning in their arguments. In addition *rapporteur* has knowledge about the syntax of arguments and argument reports and methods to provide a mapping between the various graphs.

It has been argued, Crowston & Malone [7] that it is undesirable for autonomous agents to take control over users of a system. In the particular context of a dialogue game this objection can reasonably be refuted, bearing in mind the huge success of other computer games, and we believe that future generations of users—exposed to computer games early in life—will be amenable to and not embarassed by the use of games for other purposes. Consequently *rapportuer* may be of interest to a wide variety of potential users, on the one hand any person wishing to pursue a coherent argument in a document with the collaborative aid of a colleague, and on the other hand social scientists interested in studying people in collaborative discourse could well find such a tool useful.

It is intended that the research presented in this paper should move forward on several fronts: the agent architecture will be used in the design and production of a more varied set of agents, *rapporteur* will be developed further with an increased argument syntax handling ability and more sophisticated reasoning methods, and a the design of a user interface will be developed and tested.

## 7. References

[1] Bench-Capon, T.J.M., Dunne, P.E.S., and Leng, P.H., *Interacting with Knowledge Based Systems Through Dialogue Games,* 11th Annual Conference on Expert Systems and Their Applications, Avignon, 1991, pp123–130.

[2] Bench-Capon, T.J.M., Dunne, P.E.S., Staniford, G., *RAPPORTEUR: From dialogue to Document,* Proc. Computers in Writing IV, University of Sussex, 1991, pp175–183.

[3] Bench-Capon, T.J.M., Dunne, P.E.S., and Leng, P.H., *A Dialogue Game for Dialectical Interaction with Expert Systems,* 12th Annual Conference on Expert Systems and Their Applications, Avignon, 1992, pp105–113.

[4] Bond A.H., Gasser L., (Eds), *Readings in Distributed Artificial Intelligence,* Morgan & Kaufmann Publishers, Inc., 1988.

[5] Chu D. A., *A Beginner's Guide to Mailboxes* draft, Logic Programming Section, Dept. of Computing, Imperial College, London, 1992.

[6] Cosmadopoulos Y. & Chu D. A., *IC Prolog II version 0.90 Reference Manual* Logic Programming Section, Dept. of Computing, Imperial College. August 1992.

[7] Crowston, K., Malone, T.W., *Computational Agents to Support Cooperative Work* Departmental Working Paper CSCW88-00W5, Massechusets Institute of Technology, 1988.

[8] Dijkstra E.W., *A Constructive Approach to the Problem of Program Correctness,* BIT, 8, 1968, pp174–86.

[9] Dunne, P.E.S., Staniford, G., *A Formal Language Basis for Studying Computational Properties of Graph-Theoretic* Document Models, Bulletin of the EATCS, No. 44, 1991, p292.

[10] Dunne, P.E.S,. Staniford, G. *CASS: A Cooperative Authorship Support System,* Proc. JISI'92, Tunis 1992, pp129–140.

[11] Genesereth M., Nilsson, N.J., *Logical Foundations of Artificial Intelligence,* Morgan Kaufman inc., 1987, Ch 13, pp307–327.

[12] Hinde R. A., *Ethology, Its Nature and Relations with Other Sciences,* Fontana, Glasgow, 1982.

[13] MacKenzie, J.D., *Question-Begging in Non-Cumulative Systems,* Journal Philosophical Logic, 1979, vol 8, pp159–177.

[14] McCleery R. H., *Optimal Behaviour Sequences and Decision Making,* in *Behavioural Ecology: An Evolutionary Approach,* Davies N. B., Krebs J. R.,(Eds), Blackwell Scientific Publications, Oxford, England, 1978.

[15] Milner R., *A Calculus of Communicating Systems,* Lecture Notes in Computer Science, Vol. 92, Springer-Verlag, 1980.

[16] Milner R., *Communication and Concurrency,* Prentice Hall International (UK) Ltd., 1989.

[17] Minsky, M.L., *The Society of Mind,* Simon & Schuster, New York, 1987.

[18] Raz, J., *Practical Reason and Norms,* Princeton University Press, Oxford, 1990.

[19] Robinson, J.A., *A Machine-Oriented Logic based on the Resolution Principle}* Journal of the ACM, vol 12, 1965 pp23-41.

[20] Simon H. A., *Rational Choice and the Structure of the Environment* Psych. Rev., 1956, vol 63, pp129–138.

[21] Staniford G., *Socio-ecological Metaphors and Autonomous Agents,* in *Computing with Biological Metaphors,* Paton, R.C., (Ed), Chapman & Hall, London, in Press.

[22] Staniford, G., Dunne, P.E.S., *Autonomous Agents in the Support of Cooperative Authorship* in *CSCW and Artificial Intelligence,* Edmonds, E., Connolly, J.H., (Eds), Springer Verlag, London, in Press.

[23] Stevens W. R., *UNIX network programming* Prentice-Hall, International, Inc., 1990.

[24] Toulmin S.E., *The Uses of Argument,* Cambridge University Press, 1958.