Replacing CASSANDRA

Iain D. Craig
Department of Computer Science
University of Warwick
Coventry CV4 7AL
UK EC

October 1992

Abstract

We criticize our CASSANDRA architecture as inflexible and incapable of supporting communications in any realistic sense. We argue for our position and propose a new agent structure.

1 Introduction

For the past few years, we have been developing the CASSANDRA architecture [2, 3, 4]. We initially developed two applications (both documented in [2]): one, an Air Traffic Control problem, the other a problem from Cognitive Science. We developed a formal specification in Z [28] of CASSANDRA [8]; independently, Velthuijsen specified it in CCS [21]. The architecture is in use at a number of sites around the world (e.g., [19]) and has been extended in various ways by other workers.

More recently, we have used our Z specification as the basis of a new implementation of CASSANDRA. The new implementation is written in EuLISP [22], and allows the concurrent execution of system components; in addition, we have built, again in EuLISP, a fully distributed version which will run on a network of processors (the network can be a wide- or local-area network). The new implementations will be described in [10].

Last year, we suggested [6] a number of extensions to CASSANDRA. The main extensions were:

- Incorporation of at least one declarative (semantic network) database.
- Inclusion and use of organizational models.
- Exploitation of introspective reasoning.
- Fuller use of speech acts.

Of these extensions, the proposal to use speech acts [27] as an inter-agent communications language was first suggested in [2]: in [6], we wanted even greater use of them. The idea of exploiting organizational information was first suggested by Fox [12], and we saw great merit in the proposal: the use of organizational information was intended to form a background to problem-solving and communication activities. The speech act model of communication requires that agents be aware of their goals, knowledge and capabilities. It also provides a theoretical basis for learning about other agents. The requirement that agents be aware of their internal states, and aims, suggested that agents in a CASSANDRA system be *computationally introspective*.

The extensions we proposed in [6] were motivated by the need to incorporate communications more fully within the CASSANDRA architecture. CASSANDRA was originally intended as a first attempt at defining a *multi-agent* architecture. As a first attempt, we concentrated our efforts on problem solving within the architecture: communications was, we felt, an issue that could initially be simplified and left for more complete treatment until a later date. Our recent experiences (documented in [10]) lead us to believe that this approach was incorrect and that communications plays a more central part in multi-agent systems than we believed.

In this paper, we present a critique of the CASSANDRA architecture. Our main reasons for wanting to replace the architecture can be summarized as:

- Poor integration of communications.
- Inflexibility.
- Poor explanation facilities.
- No learning from experience.
- No representation of the self or of other agents in the system.

The origins of these criticisms can be found in [7] where we argued that communication entails the formation of *expectations* about the speaker and audience, and that effective communication can only be based on models of both speaker and audience. In terms of multi-agent systems, the argument in [7] leads to the idea that each agent must possess a model of itself *as well as* models of the other agents with which it interacts. Furthermore, it can be argued that the CASSANDRA model of communications (in common with many other DAI and multi-agent systems) relies on an *absolute* concept of meaning: messages are given *a priori* meanings which all agents must respect—indeed, all agents must use the absolute meaning *precisely* so that they can communicate[1].

Having presented this critique in some detail, we will propose a new model for a multi-agent system. The new model, we believe, better satisfies the requirements

[1] This is intended as entailing that, if agents do not use the absolute meaning, they cannot interact in any way—communication becomes impossible.

that we stated above as failures with CASSANDRA. The new model is one that exploits parallelism. In [5], we suggested some ways of turning CASSANDRA's level managers into collections of parallel processes. Because of CASSANDRA is based on the blackboard architecture, parallelization turned out to be an extremely difficult process. The reasons we wanted concurrency within level managers were:

- Increased speed.
- Increased responsiveness to communication events.

In the new model, we adopt a parallel architecture because it provides greater *flexibility* within agents. Furthermore, as will become clear, intra-agent concurrency allows agents to exploit their knowledge in different ways. Of course, concurrency of this kind makes agents less prone to catastrophic failure, and may increase speed: these reasons are not of any great importance to us at present. We will use our HOLS system [9] to illustrate and partially justify the new model.

Acknowledgements

First of all, I would like to thank my wife Margaret for her constant encouragement, and for listening to what must seem endless arguments about the issues discussed in this paper; the HOLS project was her idea. Secondly, I would like to thank my research student Gary Craven for a number of helpful observations on communication and agency.

2 A Critique of CASSANDRA

In this section, we present our critique of CASSANDRA. The critique is based on the failings we outlined in Section 1. We will address the failings in the following order:

1. Inflexibility.
2. Poor explanation facilities.
3. No learning from experience.
4. Poor integration of communications, and
5. No representation of the self or of other agents in the system.

We group the first three points together and will return to them in subsequent sections.

In this section, we assume that the reader is already familiar with the CASSANDRA architecture.

2.1 Inflexibility, explanation and learning

CASSANDRA is a distributed problem-solving architecture that is based on the blackboard architecture. The blackboard architecture is, essentially, a rule-based structure. The application of rules (in the form of knowledge sources) is the method by which both types of system solve problems. If a knowledge source is unable to respond to a situation, it cannot contribute to the problem-solving process. Although rule-based systems are flexible to a certain extent, they all share this fault: this fault is often referred to as *brittleness*. Flexibility in rule-based systems comes from their ability to combine rule applications in ways that suit the problem under attack. In CASSANDRA, flexibility comes from the generality of knowledge sources and their ability to respond to a variety of contexts; in addition, the order in which knowledge sources are executed will depend upon the nature of the problem and the state of the solution process. Brittleness is a problem in CASSANDRA because it is impossible to provide a level manager with a complement of knowledge sources able to solve every possible problem which the level manager might be set. CASSANDRA inherits many, if not all, of the problems associated with rule-based systems.

In common with blackboard systems, sophisticated control is possible within CASSANDRA. This increases the range of responses that a level manager might make, but the range is still constrained by the situations that were forseen by the system builder.

When a knowledge source is unable to respond to a situation in its level manager's local database, it is not executed. When a knowledge source has only part of the information it requires available to it, the results that it generates degrade (it can also make a guess). Collectively, when the knowledge sources in a level manager encounter a local situation which is different from what they were designed to expect, they may either fail to make progress or else the quality of the solution degrades. In the case in which a knowledge source does not respond (because the triggering context is slightly different from what it expects), solution degradation also occurs. Degradation can be in the form of a deviation from the optimal solution path, or there may be a decrease in solution quality. In undocumented experiments with CASSANDRA-II and WordSys, we adjusted various parameters and observed solution degradation and deviation from the optimal solution path; we also removed knowledge sources and altered their content with the same result. In addition, in the distributed processing environment in which CASSANDRA operates, messages may not be sent as a result of this kind of failure: this, again contributes to failure, but, this time, of a non-local kind.

What CASSANDRA does not do is to engage in explanation. Explanations can be produced for users or for problem-solving purposes. The second role for explanation is discussed extensively by Schank [26]: this is the aspect we focus on here. The internal explanation process, it can be argued, is important in determining why an anomaly has occurred. The anomaly might be that information is missing, that the available information is not what was expected, or that the available information represents something unexpected. In each case, an intelligent agent should determine the causes of the anomaly and find a way of coping with it. In other words, explanations direct the agent's problem-solving activities by assisting the agent in determining new problem-solving moves in the presence of anomalies. The explanation process also has the potential of helping an agent to take reasonable actions in response to abnormal events. Schank argues that explanation is a fundamental process in cognition, and we agree with him to a large extent.

Explanation, at least in the form discussed by Schank, is related to learning. CASSANDRA does not learn from experience. When a problem is presented to a CASSANDRA system a second time, the system will again solve the problem from scratch, even though it *could* make use of its previous solution. This problem is common to all rule-based systems: they always solve every problem from scratch. Furthermore, if a new problem is similar to a previously solved one, CASSANDRA does not adapt the previous solution to the new problem. If CASSANDRA could do this, it would avoid large amounts of work. In general, however, no two problems are exactly the same: they differ in various ways. This is the reason for *adapting* previous solutions. Explanation occurs when a previous solution has some flaw or when a flaw is detected while adapting a previous solution. By explaining the flaw, the system can learn to avoid that type of failure in the future: this is the essence of part of the way Hammond's CHEF [17, 16] case-based planner operates.

2.2 Integration of communications

In the original CASSANDRA design [2], we described a communications architecture in terms of channels and ports, but did not investigate a protocol or language for communications. We assumed that we would be able to do this at a later date and integrate communications with the level manager structure in a relatively straightforward manner. Moreover, we did not investigate the implications of communications within a multi-agent system.

Our first experiments with CASSANDRA relied on an *ad hoc* set of messages. In CASSANDRA-II, there was only one message type: when a message arrived at a level manager, its contents were simply posted on the level manager's local database. In WordSys, we initially employed three message types:

- A **NEW** message. When a level manager received a message of this type, it added the contents of the message to the local database as a new entry.
- An **ADD** message type. When a level manager received a message of this type, it extracted information from the message and updated an entry in the local database. The message specified an entry in the local database and an attribute-value pair. The attribute-value pair was added to the entry in the local database.
- A **MOD** message type. When a level manager received a message of this

type, it extracted information from the message and updated an entry in its local database. The information extracted was identical to that extracted from **ADD** messages.

In the EuLISP implementation of CASSANDRA [10], we added three more message types:

- A **STOP** message type. When a level manager receives a message of this type, it immediately terminates, perhaps sending **STOP** messages to other level mangers.
- An **ERROR** message type. Messages of this type signal errors: the specific error is specified by the message's contents. A variety of responses is possible when a level manager receives a message of this type.
- An **INFORM** message type. Messages of this type had two uses within the system. The first use was to inform each agent of its identity, network address and capabilities. When used in this mode, an agent that was responsible for starting and configuring the system sent these messages to other nodes. The second use was to inform the console process of what each agent was doing at any time. The second use developed from the need to engage in remote testing of the system and also had the advantage that the user could see what was going on (this was useful when demonstrating the system to others).

In two cases, it is necessary for the sending level manager to have knowledge of the contents of local databases belonging to other level managers: it must know the name of an entry, and it must know about the contents of that entry. In the case of **ADD** messages, it must know that the attribute-value pair it is to send is not already present in the entry. In the case of the **MOD** message, it must know that the attribute of the attribute-value pair it is to send is already present in the entry. This scheme clearly makes very strong assumptions about agents: in particular, that it is possible for one agent to have reliable information about the contents of another's local database.

Velthuijsen adopts a similar approach for his BLONDIE-III system [30]: he has said[2] that he sees a trade-off between this approach (which he refers to as the 'Structural Organization' approach) and approaches that rest upon weaker assumptions. While accepting Velhuijsen's argument, we believe that the approach we adopted for our CASSANDRA implementations and which Velthuijsen adopted for BLONDIE-III[3] is far too restrictive to be completely general; in addition, it poses problems when a system is composed of a heterogeneous collection of agents. In addition, we

[2] Personal communication, June, 1992.

[3] As Velthuijsen notes in his thesis, BLONDIE-III is, in many ways, similar to CASSANDRA, and part of its design was based on our architecture: the most significant difference is that BLONDIE-III has nodes which are architecturally complete blackboard systems.

believe that the approach we have so far adopted marginalizes communications and turns it into an additional and, to a certain extent, redundant agent component.

In a multi-agent system, communications perform a number of functions. They are used to communicate useful information—information relevant to an agent's task repertoire[4]. It can also be used to convey state information: i.e., messages can convey information that assists the receiver in making inferences about the sender's internal state. Finally, messages may be used to elicit various kinds of behaviour from the reciever (e.g., control messages and messages that specify tasks).

The scheme used in the various CASSANDRA implementations conveys very little information other than that directly related to the current task. The current scheme does not convey information about a level manager's knowledge, it state or its current activities. The scheme could be extended, but its emphasis is on problem-solving messages. As we argued above, the scheme that has been adopted for CASSANDRA to date makes strong and limiting assumptions.

We initially proposed the use of speech acts in [2]. We selected the speech act model because it was principled and is applicable in a great many contexts. In [6], we expanded our argument and argued that the very fact that one agent has sent a message to another is significant (i.e., a great deal of useful inferences about the sender can be made of the basis of receiving a message). Furthermore, we argued that speech acts allow both sender and receiver to make inferences about the other party: inference can be based on the speech act itself (which we assumed, for simplicity, to be *explicitly* encoded in messages), on the content of the message, or on a combination of both. These arguments imply that communication is a rich domain which can be exploited for a variety of purposes.

Recently, we have examined the implications of the speech act model. We arrived at a structure similar to that described in [1]. The implications of this exercise are that communications must be integrated with the central functions of an agent. At the very least, the modules responsible for producing and interpreting messages must have access to information that relates to the agent's activities, knowledge and capabilities. Furthermore, messages are usually produced in order to achieve some effect: this implies that message sending is *intentional*. This has the consequence that an agent sends a message as a product of some form of decision-making process: that process depends, in part, upon the agent's current focus of attention and on its current actions. Message sending, therefore, requires access to the current goals, amongst other things. The argument in [1] is that speech act production is an intentional process: we add to this the observation that the intentionality has its roots in the agent's current activities and long-term aims. The almost rigid separation between communication and problem solving that was adopted for CASSANDRA seems to break down under this argument.

There is, moreover, an aspect to communication that we have not considered

[4] Note that we talk of a 'repertoire' of tasks, not a single problem. We believe that the agents in a multi-agent system must be capable of doing many things, sometimes concurrently.

before. Not every message an agent receives is necessarily immediately comprehensible. Indeed, in order to extract as much information as necessary from a message, an agent must be able to *explain* the message with reference to what it knows about the sender. In the case of a message that is not immediately comprehensible, it may be necessary for the receiver to fill in missing information. A message that introduces a new topic, for instance, may not immediately be comprehensible to the receiver, particularly if the receiver believes that the sender has no interest in, or knowledge of, the new topic. Until an adequate explanation has been produced, the receiver may not be able to respond in a reasonable fashion. Understanding a message about a recognized topic, as we noted, also requires explanation: part of the explanation will be based on what the receiver knows about the sender. When sending a message, the sender has a topic, a content and at least one receiver in mind: the way the message is phrased and the precise content may be determined by what the sender knows or believes about the receiver—this can also be viewed as a form of explanation.

In summary, we now believe that there is an intimate relation between the problem-solving (or central) components of an agent and those components dealing with communications. We also believe that flexible communications involve an understanding process that is based, in part, on explanation. These new beliefs are at variance with the approach we adopted for CASSANDRA. As we hinted in [9], we now view communication as providing access to other memories and agents: that is, communication is a mechanism that allows an agent to access more resources than would be available if the agent were not to engage in acts of communication. Furthermore, it is essential that agents *learn* from acts of communication in a variety of ways.

We argued above that agents are better placed to solve new problems if they remember and adapt solutions to previously solved problems. In a similar way, agents in a multi-agent system can make use of the previous experience of other agents when solving a new problem. Also, the contents of messages, and the understanding process we have argued for both afford agents valuable opportunities to learn: they can learn (via explanation and other processes) that (or how) another agent can help in solving a problem; they can learn new solutions by asking for a solution outline that can be adapted; they can also make use of accounts of the experience of other agents in generating new solutions (we briefly discussed this in [9]). The structures needed to perform some of these kinds of learning are discussed in the next subsection.

2.3 Representation of agents

In order to communicate, one must have something to say and a means to say it. One needs more than this, however. One needs to have an idea of what the audience is like. It is pointless to ask someone to do something that they cannot do: a request of this kind might be made if one is ignorant of the other person.

One reason for the inclusion of models of other agents is that an agent can make use of this information when engaging in acts of communication. For example, this information can be used to determine which other agent is likely to be best placed to answer a question or to perform some task. Such a model can also be used to inform an agent about what others are doing and their probable availability to perform other tasks. What is more, agent models can be used in the process of engaging in communication. People constantly use their beliefs about others when communicating: linguistic register, message content and grammatical aspects (e.g., use of subjunctive or conditional moods) may alter depending upon who the target of a message is. Some topics may be taboo for some people: when we know this, we can avoid those topics in conversation.

Models of *other* agents are only part of the story: each agent needs to have a model of *itself*. This is so that it can make use of its models of the other agents, and so that it can determine facts about itself for use in communication. For example, if asked to do something, an agent that has no model of itself may not be in a position to determine whether the task is possible; when it has a self model, it can answer such a question because the self model will contain information about its current tasks, knowledge, and so on. The self model can also be used to answer "what if?" questions before engaging in communication: it can ask itself questions about how it would react if it received such a message. This example shows that self models can be used in the generation of messages. It can also be used in the understanding of messages: this process relies upon an assumption that the other agents are, in some sense, similar to the receiver. Furthermore, the self model can be used to explain information in messages: the model contains information that can fill in missing information in a message—in particular, assumptions made by the sender about the receiver. Moreover, when a message is understood, the receiver's self model can be used to decide whether the message has a useful content (in the case of messages that inform the receiver), is possible (in the case of messages that request or order the receiver to do something), and so on.

The use of models of the self and others can form the basis of communication that obeys Grice's maxims [15]. For example, the self model can be used to determine a need. A message is created that is to be sent to another agent. The model of the agent that will receive the message can then be employed to adjust the message so that it says exactly what is necessary and in the right way. Other uses of such models have been suggested in [7]. The collections of models possessed by an agent also appear to contribute to a more flexible communications scheme: that is, one which avoids the absolute meanings we criticized above (the argument is complex, but, in essence, entails that meanings are not fixed but adapted by the community of language users in a way similar to that suggested by Putnam [23]).

In multi-agent systems, each agent needs to know something about the agents with which it communicates: Gasser's MACE system [13, 14] includes facilities for representing other agents—the representations in the literature are all static and relatively simple, however. Our view of models is very much more complex and dynamic. The models are updated when communication occurs and when some task has been accomplished; the models can also be updated when an agent acquires new knowledge. From the arguments given above, it can be seen that the collection of models—what might be called the agent's "social representation" is a central issue for multi-agent systems. CASSANDRA has no provision for models: they would be another bolt-on feature.

3 The New Model

In this section, we describe our new approach to multi-agent systems. The new model contains the features that were missing from CASSANDRA. We now tend to view CASSANDRA as an essentially *problem-solving* architecture for multi-agent systems, and one that excluded many important features in favour of problem solving. Although problem solving is the primary function of a system, the other features are crucial in supporting the primary function. The new architecture (for which we do not, as yet, have a name) is based upon our work on holiday planning [9]. In [9], we describe a multi-agent system whose aim is to plan holidays: the agents in the system are to produce a collective plan so they can all go on holiday together. The system has to perform the planning task, but it also has to engage in negotiation; furthermore, new agents can be introduced and old ones removed, so the environment can be relatively dynamic.

The new model is shown diagrammatically in Figure 1.

3.1 Basic structure

The new model is composed of a number of autonomous, communicating agents. The core of each agent in the new model is the problem-solving component. This is the part of the agent that solves problems (holiday planning in the case of HOLS). Because of the dynamic environments that we wish to work in, the problem-solving component must be flexible and must be able to use previous experience in performing its various tasks. We view each agent as being able to perform a number of tasks so that it can flexibly respond to the environment.

For reasons that we gave in the last section, the problem-solving component in the new model is a Case-Based Reasoning (CBR) system [25, 26, 17, 24, 20]. CBR, we believe, provides both flexibility and the ability to re-use previous experience in solving problems. In addition, CBR is a way of constructing and understanding systems that learn [25, 26].

The central component of a Case-Based Reasoning system is a dynamic memory: this memory is indexed in a variety of ways, allowing retrieval of episodes based on different factors or attributes. When an episode has been retrieved, it can be adjusted to form a new solution (the new solution, when judged adequate, can be

Figure 1: The structure of an agent in the new model.

stored in memory—a process of learning by remembering). A number of processes operate on the central memory: indexing, retrieval, and adjustment processes, for example. Most implementations of CBR systems to date have been sequential, but there is no reason why the processes that operate on memory should not execute in parallel.

The new model will be composed of a number of concurrently executing processes. The memory is the central resource in the problem-solving component, and processes that operate on it execute in parallel (the memory is protected, so that concurrent updates do not cause problems). CASSANDRA was based on an inherently sequential model, and in [5], we briefly discussed some ideas on parallelizing it. The new model starts out with concurrency. The reason for wanting concurrency is not to increase speed: we want it because it increases *flexibility*. That is, we see the need for agents in the new model to perform a repertoire of tasks: by adopting a concurrent model, we believe that we can install new tasks with greater ease.

In addition to the processes already mentioned, the problem-solving component also performs explanation. As we have argued, explanation is an important process, both for internal and for external activities. In the context of the new model, explanations can be of messages, activities and knowledge. The new model integrates explanation with the problem-solving process in ways similar to those suggested by Schank [26].

3.2 Communications

In the new model, communications facilities are greatly enhanced. Central to the communications structure are the models discussed in the last section. The models contain general as well as specific information. Information about other agents is derived from information that is supplied *a priori* (in a way reminiscent of MACE), and is also derived from messages. The self model is also based in part on *a priori* data, but it is altered at runtime by processes that inspect the activities of the agents problem-solving processes: in other words, the self model is constructed, in part, from introspection.

The two sets of agent model—the model of the self and the models of the other agents—are both CBR systems. CBR systems typically operate on episodes or cases. The dynamic component of the models of other agents is based, mainly, on the messages an agent receives: messages are the cases used in these models. Explanation facilities and learning mechanisms are clearly important in the models. As messages arrive from other agents, their contents need to be understood so that the receiver can perform appropriate actions. But messages also need to be understood so that the receiver can update its model of the sender and so that the messages can be integrated with the rest of the model. The model also serves to provide an agent with information (which can be, at best, only plausible) on the state of the other agents with which it communicates. We see the models as serving a number of purposes within the new model. The fact that the various models are CBR systems gives the new model a uniformity that CASSANDRA might have lacked. Each model (including the self model) contains information about goals, desires, knowledge, state and possible actions: these, we believe, are the main indices that will be used.

The communications mechanisms and, in particular, the protocols still need to be determined. Given the structure of the system, speech acts may still provide an adequate basis for inter-agent communication. We say this despite the criticisms made by Suchman [29]: she argues that speech acts, like conventional views of planning, provide only a *post hoc* justification of actions. We still believe that speech acts are the most principled account of communication, and that they provide the greatest coverage.

4 Conclusions

In this paper, we have presented a critique of our CASSANDRA architecture. We argued that CASSANDRA is inflexible and unrealistic in many ways: in particular, it is unable to profit from previous experience, it cannot explain its actions, and it cannot adapt to new situations. We also argued that the communication structure in CASSANDRA was, in many ways, flawed, and that the extensions to the architecture that we have proposed [6] cannot be smoothly integrated with the architecture as it stands.

We went on to propose a new model for a multi-agent system. The new model is based on CBR techniques, and we argued that CBR has a number of advantages over rule-based systems. The new model provides agents which have concurrently executing components, and which communicate to solve problems. Each agent in the new model is also equipped with dynamic models of other agents and of itself. As we argued in Section 2, these models are important to the communications process and for the correct functioning of agents.

We intend to explore the new model in the context of a holiday planning problem (see [9] for more information). The planning problem is, we have argued, one which is often performed by a group of communicating agents. Unlike many other tasks, holiday planning brings to the fore questions about the agent's selfhood and its perceptions of other agents, as well as goals, preferences and desires. Certainly, it is possible to build the HOLS program as a single agent, but the extension to a multi-agent is not only natural, but raises a number of important questions (in addition, "One only acquires an identity in a group"[5]).

References

[1] Cohen, P. and Perrault, R., Elements of a plan-based theory of speech acts,

[5] This remark was reported to us by G. Craven: he was reporting a conversation he had recently had and found this remark interesting.

Cognitive Science, Vol. 3, pp. 177-212, 1979.

[2] Craig, I. D., *The CASSANDRA Architecture*, Ellis Horwood, Chichester, 1989.

[3] Craig, I. D., A Distributed Blackboard Architecture, *Proc. AAAI Workshop on Blackboard Systems Implementation Issues*, Seattle, WA, USA, 1987.

[4] Craig, I. D., An Overview of CASSANDRA-II, *Kybernetes*, Vol. 16, pp. 223-228, 1987.

[5] Craig, I. D., *Making CASSANDRA Parallel and Distributed*, Department of Computer Science, University of Warwick, Research Report 180, 1991.

[6] Craig, I. D., *Extending CASSANDRA*, Department of Computer Science, University of Warwick, Research Report 183, 1991.

[7] Craig, I. D., *Meanings and Messages*, Department of Computer Science, University of Warwick, Research Report 187, 1991.

[8] Craig, I. D., *The Formal Specification of Advanced AI Architectures*, Ellis Horwood, Chichester, 1991.

[9] Craig, I. D., *Where Do You Want To Go On Holiday?*, Department of Computer Science, University of Warwick, Research Report 227, 1992.

[10] Craig, I. D., *The New Implementation of CASSANDRA*, Department of Computer Science, University of Warwick, *in prep*, 1992.

[11] Craig, I. D., *A Case-Based Reasoning Toolkit That Affords Opportunities For Parallel Execution*, Department of Computer Science, University of Warwick, *in prep.*

[12] Fox, M. S., An Organizational View of Distributed Systems, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 11, pp. 70-80, 1981.

[13] Gasser, L., Braganza, C. and Herman, N., MACE: A Flexible Testbed for Distributed AI Research, in Huhns, M. N. (ed.), *Distributed Artificial Intelligence*, Pitman, London, 1987.

[14] Gasser, L., Braganza, C., and Herman, N., Implementing Distributed Artificial Intelligence Systems Using MACE, *Proc. Third IEEE Conference on Artificial Intelligence Applications*, pp. 315-320, 1987.

[15] Grice, H. P., Meaning, *Philosophical Review*, Vol. 67, 1957.

[16] Hammond, K. J., Case-Based Planning: A Framework for Planning from Experience, *Cognitive Science*, Vol. 14, pp. 385-443, 1990.

[17] Hammond, K. J., *Case-Based Planning*, Academic Press, Boston, 1989.

[18] Hayes-Roth, B., Blackboard Model for Control, *Artificial Intelligence*, Vol. 26, pp. 251-322, 1985.

[19] IEATP Project 1325, *Consensus – Methodological Issues in the Design of Parallel Cooperative Knowledge Based Systems*, BAe (Military Aircraft) Ltd., University of Durham, Cambridge Consultants Ltd.

[20] Kolodner, J. L. and Simpson, R. L., The MEDIATOR: Analysis of an Early Case-Based Prolem, *Cognitive Science*, Vol. 13, pp. 507-549, 1989.

[21] Milner, R., *Communication and Concurrency*, Prentice Hall, Hemel Hempstead, 1989.

[22] Padgett, J. A. et al., *EuLisp Definition Version 0.7.5*, School of Mathematics, University of Bath, 1992.

[23] Putnam, H., The Meaning of "Meaning", *Mind, Language and Reality, Philosophical Papers Volume Two*, pp. 215-271, CUP, 1975

[24] Riesbeck, C. and Schank, R. C., *Inside Case-Based Reasoning*, Erlbaum, Hillsdale, NJ, 1989.

[25] Schank, R. C., *Dynamic Memory*, CUP, 1982.

[26] Schank, R. C., *Explanation Patterns*, Erlbaum, Hillsdale, NJ, 1986.

[27] Searle, J. R., *Speech Acts*, CUP, 1969.

[28] Spivey, J. M., *The Z Notation: A Reference Manual*, Prentice Hall, Hemel Hempstead, 1989.

[29] Suchman, L., *Plans and Situated Actions*, CUP, 1987.

[30] Velthuijsen, H., *The Nature and Applicability of the Blackboard Architecture*, Ph. D. Thesis, Department of Computer Science, Limburg Univerity, Maastricht, The Netherlands, 1992.