A NEW INTREPRETATION OF THE BLACKBOARD METAPHOR

Iain D. Craig

Department of Computer Science
University of Warwick
Coventry CV4 7AL
UK
email: idc@dcs.warwick.ac.uk

ABSTRACT

This paper describes a new interpretation of Newell's blackboard metaphor of group problem-solving activity. The new interpretation considers the experts of the metaphor to be independent, concurrently active, agents that communicate by posting information on a shared memory (the blackboard). The blackboard in our new intepretation is expanded in functionality and is implemented as a concurrently executing process. The blackboard is able to perform tasks such as agent creation, message direction and forwarding, and message censorship. We outline all of these functions and compare them with the conventional (HEARSAY-II derived) interpretation of the metaphor. We end by indicating some of the various ways in which the new intepretation can be used to construct systems that are distributed as well as parallel; as part of this, we show how the blackboard structures of the new interpretation can be recursively composed (embedded) to produce complex systems built from simple components.

1 INTRODUCTION

The logical blackboard model (Brogi, 1991; De Bosschere, 1993) has recently developed from two sources: logic programming and the blackboard architecture (Craig, 1988, 1991a, 1992; Engelmore, 1988; Nii, 1986a, 1986b). Apart from the fact that the logical foundations of the new variety of blackboard system are much clearer than those of traditional blackboard systems (a point which becomes abundantly clear when the size of a semantics of a traditional blackboard system is considered—an example of a semantics can be found in (Craig, 1991)), logical blackboards are intended as *concurrent* systems. Indeed, some of the earliest publications describing logical blackboard systems are explicitly concerned with the construction of *concurrent* logic programming systems based on a shared memory model.

The logical blackboard model consists of a central, shared database (working memory or *blackboard*) and a collection of *knowledge sources* which interact via the blackboard. Each knowledge source is an independent process that sends messages to and receives messages from the blackboard. Unlike the traditional blackboard architecture, the logical blackboard architecture does not require that the central database be constructed as a partially ordered collection of sub-databases, called *abstraction levels*. In some interpretations (e.g., Schwartz, 1993), the blackboard is divided into abstraction levels, but they are considered to be unordered, and abstraction plays no part in the functioning of the system. Furthermore, knowledge sources in logical blackboard systems are not structured as they are in the traditional interpretation of the architecture. The traditional interpretation of knowledge sources is that they are at least bi-partite in structure: they are composed of a precondition and an action. The precondition must be satisfied before the action can be executed to change the blackboard state. The precondition typically monitors the blackboard for significant changes: it is these changes that enable the knowledge source's action. Knowledge sources in most logical blackboard systems are much closer to *theories* in the logical sense. They contain facts and inference rules; they can also send information to the shared blackboard and can receive information from it. Unlike the traditional interpretation of the blackboard architecture, logical blackboard systems typically have no notion of centralised control: in a traditional blackboard system, one of the most important components is the *scheduler* or central control component. Control in logical blackboard systems is much more distributed than in the traditional case.

In this paper, we will describe a new interpretation of the blackboard metaphor. The design is currently being implemented in EuLisp (Padget, 1993) on a network of Sun workstations. The design we present below actually has a history that is longer than logical blackboard systems: we have been considering some of these ideas for approximately ten years, having previously experienced problems with the traditional blackboard architecture (see Craig, 1987, for hints and information). More recently, we have experimented with some of the concepts we now employ in our work on the ELEKTRA production system (Craig, 1991b, 1993).

Our aims in presenting the new design are as follows:

- To outline an alternative interpretation of Newell's original blackboard metaphor (Newell, 1962). The new interpretation, we believe does the metaphor more justice than the line of development that started with HEARSAY-II (Engelmore, 1988; Erman, 1975).
- To explore reflective and introspective processing in a concurrent problem-solving context. The context can either be considered to represent parallelism within an agent or a group of agents who are able to form and articulate beliefs about themselves and others in a public form.
- To provide a highly modular, yet structured architecture to support concurrent and distributed problem solving as well as research into cognitive architectures.

Whether one views the architecture as being a model of group problem-solving activity or as a collection of concurrently executing subsystems within a single individual makes for a change of perspective that can be more or less illuminating. For the rest of this paper, we will sometimes consider the system as a model of group activity using a shared resource, sometimes as just a piece of software, and sometimes as a model of a collection of agents within a single physical being. Where confusion is possible, we will make clear what our position is.

The paper is organised as follows. In the next section, we describe the architecture and relate it to Newell's original metaphor. Following the exposition of the basics of the system, we will describe how reflection and self-modelling enter the picture. We will argue that considerable amounts of power can be derived from these additions. Finally, we examine some of the possible configurations for systems constructed using the new interpretation: these configurations are quite natural and bear close resemblances to the *mirror worlds* work of Gelernter (1992). We will consider reflection and introspection in the new architecture elsewhere: we do not want to burden the reader too much.

Acknowledgements
The author would like to express his thanks to Patricia Charlton of the University of Bath for suggesting that the ideas presented in this paper should be committed to paper and for lending a copy of the workshop proceedings edited by De Bosschere *et al.* (1993). I would also like to thank my wife Margaret for helping me with the typing of the manuscript, thus preventing me from further aggravating my RSI.

2 A NEW INTERPRETATION OF THE METAPHOR

The original blackboard metaphor (Newell, 1962) considers a group of experts who are jointly solving a problem that cannot be solved by a single individual. Each expert brings specialist knowledge to the problem. The experts solve the problem by writing on a physical blackboard. They may write and draw things on the blackboard and may also read what is present. The problem is solved when the experts collectively decide that it is solved.

The blackboard metaphor is clearly about group problem-solving activity. In the metaphor, the experts observe what the others write on the blackboard and responds either to single items that have been written there or to collections of items that appear on the blackboard. It is important to note that experts can notice *individual* items as well as *collections* of them: experts can respond to simple information as well as complexes. Part of the problem-solving behaviour that is exhibited by the agents is concerned with connecting things together on the blackboard. While one expert is writing or drawing something on the blackboard, the others can be thinking about what they have seen or working on some part of the problem—the problems solved by such groups are assumed to be such that they can be decomposed into relatively independent sub-problems. The behaviour of the experts is clearly *concurrent*: they do things while other experts are active.

2.1 The Conventional Interpretation

The metaphor is restricted somewhat in the conventional interpretation of the metaphor (Craig, 1988, 1991a, 1992; Engelmore, 1988; Nii, 1986a, 1986b). The experts are active sequentially: access to the blackboard is sequential. No two experts are able to access the blackboard at the same time; when one expert is active, the others are typically inactive. There is also a central scheduling or control mechanism that mediates all activity by the experts. The scheduler implements control schemes that determine which experts become active and when. The scheduler, to a good approximation, controls access to the blackboard. Furthermore, in a real group problem-solving session, experts will work on the blackboard by *erasing* as well as adding items; in the conventional interpretation, erasure is typically absent (though see (Craig, 1989) for an approach to deletion) because it introduces, *inter alia*, problems of non-monotonicity and dependency analysis.

In the conventional interpretation, experts are seen as relatively straightforward sequential *knowledge sources*. Internal control in a knowledge source is usually simple, amounting to little more than the use of conditionals and iteration inherited from the implementation language (at least, this is the case in many LISP implementations).

Nii (personal communication, 1985) has informed us that CAGE (Nii, 1986c) was used to experiment with knowledge sources that exhibited somewhat richer internal control: specifically, knowledge sources were permitted to use meta-knowledge to control the execution of the their component rules. Such a use of meta-knowledge was purely local in nature and extent and could not guide problem-solving activities at a more global scale (Nii, *ibid.*, claimed that meta-knowledge must be no more than local in scope). Even with these extensions, CAGE knowledge sources remain unable to engage in strategic, tactical and planned behaviour. One reason for this is that CAGE is based on the conventional interpretation of the metaphor, and, consequently, its knowledge sources are really sequential in form (CAGE allows the user optionally to have knowledge source actions execute concurrently, but control at the finest grain remains sequential). Knowledge sources cannot rely on purely local information (i.e., information which is entirely stored within themselves) between invocations. That is, they are forbidden to retain state information between activations or invocations: all state information is, at least conceptually, maintained on the global blackboard and is, therefore, open to modification by other knowledge sources. Once a knowledge source instance has executed, all of the information contained and manipulated by that instance is, in effect, lost (it could be reconstructed from the blackboard and a history of events—such a history is made available by the Blackboard Control Architecture of Hayes-Roth (1985), but is computationally expensive to extract).

Within relatively conventional systems such as CAGE, knowledge sources are of a particular kind: this kind is not really implied by the blackboard metaphor, only by the needs of a certain way of implementing the metaphor as a working program. The fundamental idea is that a knowledge source is instantiated in accordance with a particular context on the blackboard; once instantiated, the knowledge source is executed to update the blackboard and the instance is then immediately deleted. The information contained within an instance is transitory. As a consequence, knowledge sources tend to be relatively inflexible in their internal behaviour; it should also be noted that there has been a tendency to reduce the size and scope of knowledge sources (a point we have previously made (Craig, 1989)): this also tends to reduce the need for internal control beyond that provided by the implementation language or by production rules (or by Prolog clauses).

2.2 A New Interpretation

For a number of years, we have been concerned that the conventional interpretation of the blackboard metaphor was, in many ways, too closely tied to a particular method of implementation (even though it can be argued that we have, ourselves, contributed to this interpretation's popularity, for example in Craig, 1988). Specifically, the conventional interpretation—what has become the "blackboard architecture"—despite many disclaimers, is founded on the sequential implementation of the metaphor in which centralised control is necessary. Central control was introduced because it is the easiest way to control the actions of the knowledge sources a system contains. We have attempted to move away from central control with our (now defunct) CASSANDRA architecture (Craig, 1989, 1991a).

We consider that there is a genuine need for the original metaphor to be reconsidered in new ways. Nii and Rice have attempted an exactly similar exercise: this resulted in the POLIGON system (Rice, 1986). Like CASSANDRA, POLIGON was relatively unsuccessful, but for different reasons—POLIGON was never able to achieve the speed increases demanded of it. Our new proposals have little to do with speed; instead, we want to re-interpret the metaphor in what we believe is a more natural fashion than the conventional interpretation.

It must be admitted that the interpretation that we propose is partly motivated by the availability of suitable technology. We have been considering an approach similar to the one described below for a number of years (since 1985, in fact), but have not proposed it for the reason that we did not have available a platform which supported concurrent and distributed processing: now we do (we have EuLisp).

It is, though, the naturalness of the interpretation that makes it appealing, not its relationship to a newly available technology or implementation techniques. We strongly believe that powerful models come first and that implementations must be found for them: this is, in many ways, similar to the original situation with the blackboard metaphor. We will continue to refer to the conventional interpretation of the blackboard architecture below in order that readers familiar with it may use its concepts in understanding our new model.

Our new interpretation retains the concepts of central blackboard and of a group of experts who are collectively solving a problem that one expert alone could not solve. The experts in the new interpretation still operate on the blackboard and still respond to changes on it: the addition or modification of information by one expert can still be monitored by other experts, and experts may also consider entire collections of items on the blackboard and to group or otherwise operate on them. Experts can, in the new interpretation, delete information that is stored on the blackboard: this allows items to be posted on the blackboard that are to be read by designated experts and it also allows competition between experts (the first expert to read the item gets the information and no other expert is allowed access to that item). The information posted on the blackboard can also serve an inhibitory role: it can forbid experts to work on some problem or to post certain classes of information. We will consider this last use in a later section.

The blackboard retains its status as the primary repository for public information. Experts are still required to watch the blackboard and to change the blackboard state in order to make public contributions to the problem-solving process.

As in the conventional interpretation, the blackboard is a public, global database that holds the results generated by the experts who are solving the current problem. There are differences between our new interpretation and the conventional one:

- The blackboard is not necessarily divided into abstraction levels or panels.
- Information placed on the blackboard by an agent is information that the agent wants to make public.
- The blackboard is a form of persistent message store amongst other things (it also serves as a database very much in the sense of a conventional blackboard database—the two functions are not irreconcilable). Part of our view of the blackboard is similar to Gelernter's LINDA tuple store (Gelernter, 1985).

We will discuss each of these points in turn.

In the conventional interpretation, the blackboard is divided into a hierarchy of abstraction levels. There be more than one hierarchy, in which case, the blackboard is divided into panels or planes, each having a different topic, but all relating to the overall problem-solving process. The abstraction hierarchy can be considered an outline plan for solving the problem (this view is reported by Nii (1986a)) or as a skeletal representation within which to describe the solution in different ways (this is a consequence of our view of representation in blackboard systems, see Craig, 1993a). In the present setting, we decided not to adopt abstraction levels because, in the context of a group problem-solving episode, they represent a *unique*, *agreed* view of the problem: that is, all experts within the group must agree that the abstraction hierarchy represents the best or most appropriate framework for solving the problem. We believe that this is overly restrictive and that such agreement may be only superficial within real groups—we suspect that such agreements may be, on many occasions, agreements more to do with group harmony than with real concord, and that in real groups, experts will maintain their own abstractions and translate between them and the publicly agreed ones as and when the need arises.

What is needed is a common vocabulary (or representation system) for external representations. This common representation is an agreed or imposed code for communication amongst experts. The style or method of encoding (representation) within an expert is not directly relevant to the common representation, with the exception that everything that can be externally (i.e., publicly) represented must be representable in the internal representations of experts. Whether translation is required to convert items in the public representation to internal (i.e., expert-specific) representation is basically irrelevant—if there is such translation, the translation process must be very rapid. Furthermore, it is necessary that we adopt a totally neutral position with respect to the translation process itself: it may be the case that atomic public items are represented internally as a complex structure (that is, it may be the case that there is not always a one-one correspondence between items in the public and internal representations).

Although we will *allow* abstraction levels and will certainly encourage the *use* of abstraction, we prefer to avoid making it a necessary component of our new interpretation. We prefer, instead, that experts maintain their own views on abstraction and the organisation of abstractions. We also prefer that experts do not become burdened with unnecessary and, perhaps, inappropriate abstractions (publicly agreed ones, that is).

The second point relates to the *use* of the blackboard by knowledge sources and experts in the original metaphor and in the conventional interpretation. The conventional interpretation requires *all* communication between knowledge sources to be implemented in terms of alterations to the blackboard. When two knowledge sources need to communicate some information, that information *must* be posted on the global blackboard in order to make such communication possible. This has the following consequences:

(1) The information thus posted is completely public in the sense that any knowledge source may inspect or update it. It is possible that a knowledge source (the poster) posts the information on the blackboard. Then the scheduler executes at least one other knowledge source before the second knowledge source reads the information put there by the poster. Note that this requires that the poster be aware of the need on the part of the second for the information—see point (3) below. During the intervening period in which other knowledge sources are executed, it is possible for any one of them to update the information posted on the blackboard by the poster. In other words, it is possible for any of the intervening knowledge sources to destroy the information being communicated, thus confounding the communication.

(2) The information to be communicated must be represented in a form that is unreadable by knowledge sources other than those that engage in the act of communication. Here, by 'unreadable', we mean that, typically, it must be named by an attribute or set of attributes that are not used by (indeed, are unknown to) other knowledge sources. This has two consequences. The first is that the representation employed on the blackboard is complicated by the addition of all these new attributes (which are, in any case, only of local interest—by design, they do not represent anything that is necessary to the global solution). The second is that knowledge sources must, in effect, be aware of which representations or attributes are used by other knowledge sources. This leads to the third problem.

(3) An assumption about knowledge sources in the conventional interpretation is that they are totally modular. That is, it is possible to add or remove knowledge sources with the only consequence that the system's performance improves or degrades as a function of the available knowledge. A consequence of this assumption is that knowledge sources must not engage in acts of communication that assume the existence of other, specific, knowledge sources. A further consequence is that knowledge sources should make no assumptions whatsoever about the information and representations that are required by other, specific, knowledge sources. In other words, knowledge sources should be totally ignorant of all other knowledge sources, their existence and their representations: they should never assume that a particular knowledge source is present in the system. The direct posting of information of the kind under consideration violates this assumption in both ways.

It should be noted that even though the modularity assumption was prominent in the HEARSAY-II literature, the actual implementation of HEARSAY-II involved direct communication between knowledge sources by means of shared variables. Furthermore, special slots were present in blackboard hypotheses for inter-knowledge source communication. The assumption in (3) above was clearly violated. Since that time, however, inter-knowledge source communication has tended to follow the norms.

What is even more important for present purposes is that the differences between internal and public representations becomes blurred in the conventional interpretation. There is, to be sure, a distinction between what is posted on the blackboard and what remains inside a knowledge source. Yet, communication between knowledge sources in the conventional interpretation of the metaphor is not confined to communication between instances of two different knowledge sources: it extends to communication between two instances of the *very same* knowledge source. In other words, if a knowledge source is to retain state information across invocations, it must post information on the blackboard. This does not necessarily violate the modularity assumption (point (3) above), but it certainly serves to make construction, manipulation and consistency maintenance considerably more difficult. (It can be argued that communication across invocations of the same knowledge source does violate (3) above because each invocation deals with a different activation context.)

The main point for us is that the assumptions underpinning part of the blackboard architecture force what is properly internal to a knowledge source to become public. Such a problematic situation is forced on us by the way in which knowledge sources are instantiated and then destroyed once they have been applied (in a sense, knowledge sources are like functions in functional programming—they hold no permanent state and are applied and then only to go out of existence).

The absence of permanent local state is what makes control *within* knowledge sources in the conventional interpretation into such a trivial matter. Since there is no permanent state within knowledge sources, the only problem that they face, once their execution begins, is which change to make to the blackboard. Knowledge sources in the conventional interpretation are unable to make global decisions and are also able only to make decisions based upon the current blackboard state (plus the triggering state which may have obtained some considerable time before execution). There is no historical information retained within a knowledge source that can be used in making decisions; any such information must be held on the global blackboard, thus incurring problems (1) and (2) above.

In summary, the conventional interpretation forbids knowledge sources from having their own histories. Furthermore, all historical information *must be made public*, even if it is properly (logically) private to a particular knowledge source. We may summarise the above arguments thus:

All information that is to be remembered must be made public.

The issue of public versus private information leads to the next point of difference between the two interpretations: our interpretation requires more of our knowledge sources, so much so that we now term them *agents*. The difference between our interpretation of experts and those of the conventional interpretation is that under our interpretation, our agents, have internal states of arbitrary complexity (hence have private information and processes) and can engage in purely internal processing that is not directly visible to other agents. An agent in the new interpretation is closer to the original notion of an expert. Agents run concurrently (and may have concurrently executing components, we think this desirable but do not require it), so that they may engage in problem solving activities in parallel with the problem-solving activities of other agents; they may also engage in problem-solving activities at the same time as other agents are manipulating the blackboard. The blackboard is no longer the central focus of the architecture: it is still there, but agents do not rely upon it for all their state information.

Agents are active the whole time the system is running. At any one time, an agent will be interacting with the blackboard, engaging in purely internal problem-solving activity or communicating with other agents. Purely internal problem-solving activity is invisible to all other agents until the agent doing it makes an explicit act of communication about it: other agents remain ignorant[1] about the internal activities of an agent until the agent tells them about what it has been doing.

Velhuijsen (1992) points out that there are two main uses of the blackboard in the conventional interpretation. The first is as a representation of the solution process' state; the second is as a communication medium. Most implemented blackboard systems combine these two functions. Indeed, we combine them in the new interpretation, but agents in our new interpretation do not rely upon the blackboard to record all state information because they are able to represent their own, local (persistent) state. Public communication is performed via the blackboard as in the conventional interpretation, but private information can also be retained and manipulated.

Our reinterpretation of the blackboard metaphor involves the amplification of the activities that experts can perform. As has been noted above, agents are able to engage in continuous processing, some of which is invisible to other agents in the system (and to observers of the system). As a consequence of this, agents possess internal states which are of a far more permanent nature than in the conventional interpretation. The internal state of an agent plays a part in determining the actions that an agent will take during problem-solving and other activities. In other words, agents in the new interpretation have a *history* and, moreover, a sense of history.

Agents possess the knowledge required to solve problems, and they also contain information about communication. They know about other agents (unlike knowledge sources in the conventional interpretation) and their specialist skills. Agents also possess meta-knowledge and are able to reflect and introspect: all the advantages of reflection and introspection are had by agents. We do not expect agents to be able to model themselves or other agents, for that will probably make them much more complex than we desire—this is *not* a multi-agent system in the conventional sense. What we allow in the new interpretation is for agents to choose the best or most suitable way for them to solve their current problem, and they can also decide whether to make a contribution and how to make it. We allow agents to employ their own, internal, control regimes in flexible ways: we also allow agents to plan if the need arises. Agents now have a considerably more power than in the conventional interpretation, but not as much as in a multi-agent system.

The blackboard database retains its central position in our interpretation. It is the place where all public information is stored: it is a shared database. Agents are allowed to inspect the contents of the database, and they are allowed to update it. Agents are in-

---

1. Strictly speaking, agents may try to figure out what is going on inside another agent, but they can only be certain of what has actually occurred when the agent in question says what has happened (assuming that the agent is truthful about its cogitations).

dependent processes, so the blackboard database is a shared data structure. Some way of controlling access to the blackboard so that the *Readers and Writers* problem can be solved. We do this by placing the blackboard in an separate process called the *blackboard process* (or just *blackboard*, for simplicity and uniformity). Agents communicate with the blackboard by sending messages. Messages from agents can be of two forms:

(1) Messages containing information to be added to the blackboard, or information that updates something already present on the blackboard. These message types correspond to those found in conventional blackboard systems.

(2) Messages requesting information from the blackboard. Information from the blackboard may be in the form of a set of objects that satisfies some predicate supplied in the request from the agent. An example of this is the `find` operation familiar from blackboard systems. `Find` returns a set of objects stored on the blackboard that satisfy some arbitrary predicate. The `find` operation is often associated with other set operations, so we provide such operations for agents to use. The blackboard itself can apply set operations to collections of objects that it stores. As will become clear, the blackboard in the new interpretation can contain information of more varied types than in the conventional interpretation: any of the information held on the blackboard can be requested by agents—this allows agents to reason about other agents, as well as themselves (even though we prefer not to *require* reflection).

The blackboard in the new interpretation has much more power than the conventional blackboard monitor. As part of its function, the blackboard still implements operations familiar from conventional blackboard monitors. In particular, the blackboard in the new interpretation performs the task of informing agents of state changes that might be of interest: this corresponds directly to the event-based triggering function provided by conventional blackboard systems (for example, BB1 (Hayes-Roth, 1985, 1986) and NBB (Craig, 1987)). In the new interpretation, agents must inform the blackboard process of the events in which they are interested: this requires that agents send the blackboard process messages that inform it of what kinds of events it should notify them of. When an event of a particular kind (say the addition of a completely new piece of information) occurs, the blackboard process determines which agents need to know of the event and sends each a message containing a specification of the event. Upon receipt of such information, agents may process it in various ways, for example determining that some action must be performed.

The new interpretation also allows blackboard entries to be destructively read. That is, the reader of an entry can (optionally) take a copy of the entry and then remove the original from the blackboard. This operation is similar to the destructive read in Gelernter's (1985) LINDA. This feature allows a number of operations to be performed in the new interpretation that are hard to perform in the traditional one and that are relatively useful. In particular, destructive read allows agents to perform the following operations:

- Agents can post entries on the blackboard that are to be read by a specified agent. When that agent has read the entry, the entry is deleted: this ensures that the blackboard does not become cluttered with unwanted data.
- Destructive read can also be used to implement private communications between agents in the sense that an entry can be marked with the name of a reader agent. No other agent is able to read the contents of that entry. Once the reader has obtained the message (entry), the message is deleted so that privacy can be maintained.
- Destructive read can be used to implement a competitive mechanism. A

message can be posted on the blackboard: the message is marked so that it must be destructively read (this could also be implemented by a distributed or implicit protocol). A set of agents is active and attempts to read the message. Because of the arbitration mechanisms implemented by the blackboard process, only one agent is able to read the message at a time. When the message is read, it is deleted from the blackboard. This ensures that only one agent will receive the message and act upon it: the other agents who tried to read the message are, essentially, locked out of the transaction and so cannot participate in the task mentioned by the message (or alternatively: cannot make use of the information mentioned by the message). Destructive read can also be used as an inhibitory mechanism: messages can contain also prohibitions.

In the conventional interpretation, deletions of entries on the blackboard are not performed. In our new interpretation, we allow destructive read as well as a mechanism to delete clusters of entries. The general deletion mechanism is allowed because the blackboard is no longer considered to be the main repository of state information: the state representation is distributed across the agents in a system.

Agents can also request the blackboard process to perform actions when events with certain, specified properties, occur. One example of such a *requested action* (or *request* for short) is to send messages to some agents when an event occurs[2]. That is, when an event of a particular kind occurs, the blackboard sends a message to a number of agents (probably informing them of the event, although the message may contain arbitrary information). This message-sending activity is caused by the blackboard process satisfying a request that was made by one of the agents in the system. The agents which are contacted in this way would probably not otherwise receive notification of the event. However, an arbitrary action can be performed as the result of a request. Requests may specify arbitrary conditions which are to be satisfied (as a precondition of the action) before the action can be performed: the condition may relate to an event, but it can also refer to the state of the blackboard database. Another class of operation that can be requested is for messages to be sent to non-local destinations (see below for details).

In certain circumstances, it is necessary only for agents to receive what amounts to a summary of what is on the blackboard. In other circumstances, some agents may not have any need to know about some kinds of information. To handle these cases, the blackboard process may be set up to summarise and censor information.

The blackboard can also inform agents that they should stop processing. Such stop messages might instruct agents to terminate or merely inform them that there is no need for their results. The form of this kind of message depends upon the application.

When an agent is introduced into the system, it must *register* with the blackboard process. We allow agents to be dynamic and to come into and go out of existence at any time: agents can create agents—a kind of reproductive behaviour—either by explicit construction or by copying. (Each agent can be thought of as an abstract data type or object in an object-oriented programming language.)

The blackboard can also contain a specification of the agents to be created (recall that agents are akin to classes of object in an object-oriented language). The blackboard can create agents by instantiating the specifications of the agents whose specifications it contains. This instantiation mechanism can be invoked by the blackboard itself as a result of some internal action or decision, or it can be invoked as a result of the reception of a message from some external source.

When an agent is added to the system by either mechanism, it must inform the blackboard process of a variety of information, some of which is obligatory, and some discretionary (whether information is obligatory or discretionary depends upon the agent

2. Reqests can be thought of as a kind of demon.

and the task to be handled). Obligatory information includes:

- The name of the agent and its location.
- The events of which the agent should be notified.[3]
- The requests the agent makes of the blackboard process.

Clearly, the blackboard process needs to know the name of each of the agents which will use it. An agent is located in a particular place: in order to communicate with an agent, the blackboard process needs to know where it is (this amounts to the agent's network address). The blackboard process needs to know the event types that are of interest to an agent in order that it can inform the agent when an event of one of the given types occurs: some agents will, of course, be interested in all events, so the specification must cater for this (as well as the case in which an agent is not to be informed of any events—this is not possible in the conventional interpretation, but may be of use to systems using the new interpretation). Requests have to be registered so that the blackboard process can honour them. An agent can register any number of event specifications and requests—this includes zero.

Discretionary information might include:

- The tasks which an agent can perform.
- The acquaintances of an agent (i.e., the agents which the registering one knows about and is prepared to make public).

Information of any kind that is stored on the blackboard as the result of agent registration is potentially available to be disseminated to other agents, should it be requested. For example, an agent may request the blackboard to inform it of the events in which another agent is interested; an agent may, equally, request the blackboard to inform it of the acquaintances of another agent, or the names and locations of all the agents that use the blackboard process.

Some agents may require that some of the information that is stored on the blackboard may only be disseminated amongst a trusted subset of the agents in the system. For example, an agent may only allow the blackboard process to tell a restricted population of the agents in the system of its acquaintances. Mechanisms are provided (the censor mechanism mentioned above) to provide such restricted access to information, and restriction information is stored on the blackboard. Agents which are not trusted with information will be refused access to information when they attempt to make requests that are forbidden to them. Highly trusted agents may have access to the information about access rights.

In the conventional interpretation of the blackboard architecture, there is a central control component called the *scheduler*. The scheduler is responsible for imposing and controlling the use of all strategies and tactics used by the system. The scheduler has information on the capabilities of the knowledge sources contained in the system and also has access to the contents of the central database (the blackboard). The new interpretation, like others (e.g., BLONDIE-II (Velthuijsen, 1992)) separates the global blackboard from the experts that comprise the system. In this organisation, the global database is encapsulated in a separate process and experts can access it only via an explicit protocol. Furthermore, the new interpretation considers experts as independent, autonomous entities which possess a private state. A scheduler, in the conventional sense, would need to have access to the state of the central, global blackboard and to the state of the various agents if it is to perform in a manner akin to that in the conventional interpretation.

A form of centralised scheduling is possible: one based on meta information stored in the blackboard could perform scheduling tasks somewhat similar to those in the conventional scheduler. However, and we believe this to be a major point, centralised

---

3. Initially, at least, we allow agents to register event notification data only when they are created. Perhaps it will be necessary to make registration of this kind of information a dynamic process.

schedulers, given their need to access relatively large amounts of information, can pose the threat of becoming bottlenecks; furthermore, centralised control may, because of communication delays, lead to inappropriate behaviours. The bottleneck occurs because the agents in the new interpretation are concurrently active: the control decisions made by a central controller will be dependent upon messages from the blackboard and from the various agents of the system which are being consulted—these messages are the responses from the various system components to information requests made by the scheduler—the load imposed by the processing of these messages may be very high. Furthermore, control decisions need, often, to be made within a very short time of the need for a decision being detected; if the controller is dependent upon sending messages and waiting for replies, a delay necessarily is incurred. This has the consequence that decisions may not necessarily be made in sufficient time. There is also the possibility that one of the agents in the system may have disappeared: this has the implication that control decisions must be made in the face of incomplete information.

The new interpretation we propose is composed of agents that are relatively powerful. As we noted above, agents are allowed to engage in their own, local control processing. Agents are able to inspect and update the blackboard *and* they are permitted to engage in direct (and private) communications with other agents. Even though we do not want our agents to be *too* powerful, we do expect them to be able to make decisions based upon a variety of factors—the state of the blackboard, recent events, the state of the non-blackboard environment. We intend the new interpretation to be closer to the situation with human agents in the original blackboard metaphor: this suggests that control should be distributed across our agents. We therefore allow agents to control themselves and to mediate their interactions via the blackboard as well as by means of one-to-one communications between themselves (that is, we now explicitly allow agents to talk directly to each other in order to, as it were, hold conversations). We intend that global control of the kind assumed by the conventional interpretation should not occur; of course, we will allow global control if absolutely necessary, but we prefer to distribute it across agents and to equip agents with various *policies*—for example, turn taking as a means of regulating interactions across the blackboard. The introduction of communications policies is, we believe, a novel aspect of the new architecture. Like all such devices, though, policies will serve only as guidelines that we expect our agents to follow: deviation from norms as well as developments of them are possible.

The general form of a system of agents and their blackboard process is shown in Figure 1.

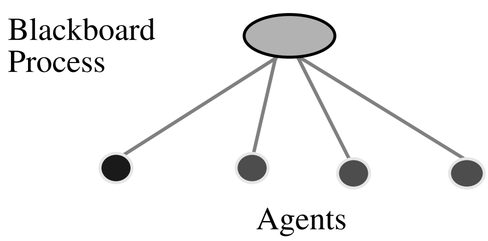

Figure 1. The basic structure of a system constructed using the new interpretation of the blackboard metaphor.

In Figure 1, the blackboard process is represented by the shaded ellipse; the agents which use the blackboard are represented by shaded circles. Communications channels between the blackboard and the agents in the system are represented by lines: the channels are intended to be bi-directional.

3 CONFIGURATIONS

We refer to a blackboard process and a collection of agents as a 'system'. Systems are to be thought of as independent entities that are able to solve complex problems using distributed control schemes. Each system will be relatively expert in some domains and will be able to solve some kinds of problem, but they are not expert in all domains, nor can they solve every kind of problem. Furthermore, each system will be relatively constrained by the processing power available to it and by local constraints on inter-process communication.

It is possible to view a system as a model of a single agent that is composed of a number of concurrently active modules that communicate via a shared working memory. Under this interpretation, a system represents a single agent: the agent being represented is of larger scope than the agent components that comprise the system. If the single agent interpretation is adopted, it can be seen that such an agent may need to communicate with other, similar agents in order to solve complex problems. In other words, the new blackboard metaphor interpretation leads to the construction of collection of individual systems, each system being an element in a distributed collection of problem solvers.

When this possibility is admitted, it can be seen that the resources of systems can be pooled in order to form larger systems of greater power. What is needed is some means of communication between systems and some way of interpreting communication. Communication can be achieved at the architecture level by allowing one or more agents to control communications with other systems. The blackboard process, in this case, maintains information about the other systems with which its system communicates. Such information needs to be public within a system in order that the system's agents be able to access it in order to engage in acts of communication with other systems. Together with low-level information such as network addresses, the blackboard process maintains information about what is known of these other systems—for example, their areas of expertise, their apparent status (are they fully occupied on a problem or do they have spare capacity? what is the apparent state of progress on a mutual problem? and so on). This information can be accessed by the agents in a system in order to determine how, what and when to communicate.

The interface agents mentioned above are responsible for co-ordinating communication with other systems. They send to and receive information from other systems and are responsible for registering information about other systems when requests come to initiate communications. As with agents, other systems must register with a system before communication can take place. This is so that the various systems can maintain information about the systems with which they communicate. The information held by a system about communicating systems is subject to the censorship mechanisms mentioned above: this is because we do not always expect *every* agent in a system to be able to engage in communications with agents outside the system (perhaps the term 'local system' would be clearer).

The model of communication that we have just described is the one familiar from multi-agent systems. We will call it the 'flat' communications configuration because all systems are arranged in a flat topology: no hierarchy is implied by the description. We refer to a collection of individual systems as a 'cluster'. The flat configuration is shown in Figure 2.

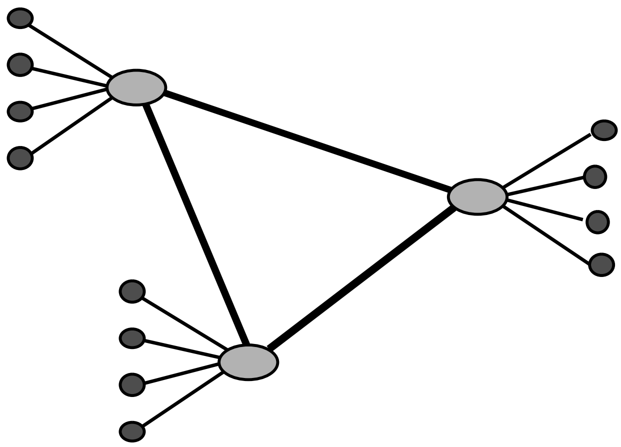

Figure 2. The flat configuration of systems. Each of the component systems is able to communicate with others in the cluster. There is no implicit hierarchy. In order to communicate, a system needs to know of the name and location of the other system.

In Figure 2, we have retained the convention we adopted for Figure 1: blackboard processes are represented by ellipses and agents by circles. Communications channels between agents and their local blackboard processes are depicted by thin undirected lines. The Figure shows three such local systems. The thicker black lines connecting the three blackboard processes represent communications channels between blackboard processes: these channels are bi-directional and will typically connect systems that are physically distributed. It is to be expected that each of the three blackboard processes shown in Figure 2 will run on its own processor: the channels between blackboard processes will be implemented by communications network services (for example, a Unix socket).

Given the flat configuration, it is possible to introduce hierarchical structure into the arrangement of systems in a cluster. There is no need to introduce special mechanisms in order to support hierarchical clusters. All that is needed is for systems to record the relationships which hold between them and their acquaintances (in the sense of Hewitt). The existence of hierarchical (or heterarchical) clusters implies restrictions on the kinds of message that can be exchanged between systems.

When a system becomes part of a cluster, it receives messages from its acquaintances and it sends messages to them. Given the distributed nature of clusters, message passing is the principal method of information exchange. When a message arrives at a system, the communications agent determines whether the information contained in the message should be posted on the blackboard so that it becomes available to the agents within the system. We anticipate that the vast majority of traffic between systems in a cluster will be of such a kind that it is posted on the local blackboard. Other information is, of course, possible: housekeeping and control information are example types—some of this information might be of general interest, of course, and we do not make any prescriptions at present about how it should be handled.

Given the relatively loose structuring of a system, it is quite possible for suitably equipped agents to belong to more than one system. That is, an agent may access the contents of and write to more than one blackboard. In such a case, the agent can participate in the processing of two (or more) systems. We refer to this configuration as the 'sharing' configuration. The additional equipment needed by an agent in order to support this kind of configuration is the ability to send and receive messages from a remote blackboard. The basic concept of the shared configuration is depicted in Figure 3: it forms the basis of the recursive embedding of systems.

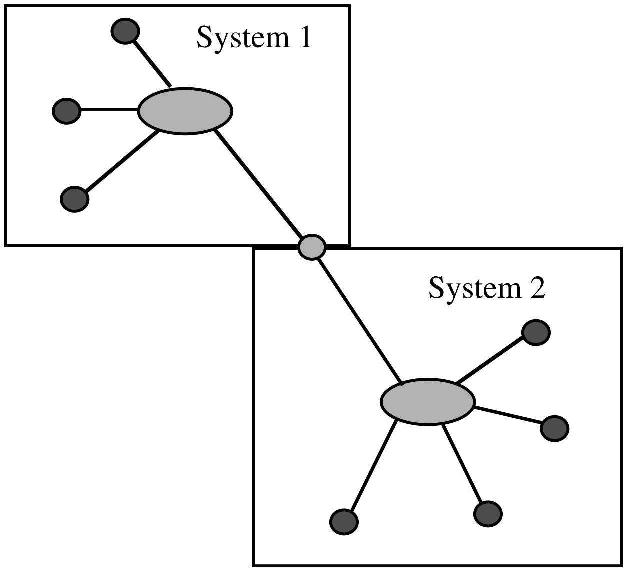

Figure 3. Two interconnected systems. The agent midway between the two systems (depicted as the cross-hatched circle) communicates with both blackboard processes and can transfer information from System 1 to System 2 and vice versa. It can also engage in purely internal processing that is dependent upon information from both these sources.

In addition to the flat and sharing configurations, it is possible to structure clusters in a recursive manner. In this structuring, an entire system looks like an agent. This allows agents to be composed of complete systems, and, therefore, to implement complex functions. A recursive configuration can be constructed from a shared configuration: an interface (sub-)agent is designated to communicate with the embedding blackboard. The interface (sub-)agent accesses the embedded blackboard and is responsible for transferring information from the embedded to the embedding blackboard. It may be necessary to engage in representation changes across this interface because, it may be assumed, the objects and structures represented on the embedded blackboard will have smaller scope and refer to external objects of smaller grain than those on the embedding blackboard.

The essential property of an embedded blackboard system is that the interface to it, as far as the embedding blackboard is concerned, is identical to that presented by an ordinary agent. From the viewpoint of the embedding blackboard, the embedded system *is* an agent. (Note that one consequence of embedded systems is that they may be treated as instantiable entities—they are, in this respect, akin to agents, and many instances of the same system type may be present in a cluster at any time.) The embedded configuration is shown in Figure 4.

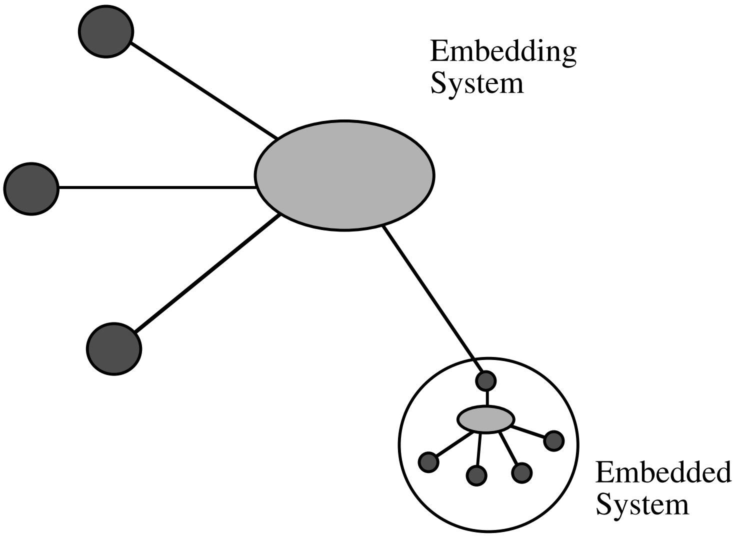

Figure 4. The embedding of one system within another. The interface between the two systems is provided by an agent that belongs to the embedded system. This interface agent is responsible for ensuring that the embedding system 'sees' the embedded one as a complete agent (and not another system belonging to the cluster).

The introduction of embedded systems leads to an interesting and important structuring principle. A system constructed using the new interpretation may be composed of embedded clusters of systems. In other words, it may be composed of a hierarchy of systems, each appearing as an agent to the next highest level in the hierarchy. Information can be passed up and down the hierarchy (and perhaps in other directions, too) in order to solve problems and perform other tasks. As one progresses down the hierarchy, the ability of systems to perform complex tasks will decrease until one reaches what amounts to a motor or perceptual level. It is also expected that the amount of meta-level processing will decrease as one descends the hierarchy. Only in the topmost levels of the hierarchy can one expect meta-level processing to be present.

Embedding also provides an analytic tool of some value: it allows processes to be collected together and viewed as being the result of the processing of a single agent—the fact that the agent is composed of a complex, interacting collection of systems is of no relevance to the analysis at *that* level. The analysis process (which is part of the *design* of systems built using the new interpretation) continues in a recursive fashion until some agreed bottom layer is encountered.

4 CONCLUSIONS

We have now presented the outlines of our new interpretation of the blackboard metaphor. Our interpretation differs from the conventional one in assigning greater functionality and power to the agents which represent the problem-solving agents of the original metaphor. The increase in agent functionality is a result of modelling them on the complete experts of the metaphor. Our experts are independently executing processes that communicate either via a shared blackboard database or directly by message passing. The blackboard is interpreted as a separate process which contains information that is publicly available to all agents in a system. The information stored on the blackboard is publicly accessible and is represented in a form that can be interpreted by all agents in the system. The agents that form the system may use the representation employed for the blackboard or may use one that is more restricted in scope: however, it is not possible for an agent to use a representation that it and only it can understand—for all uses of representation that leads to communication, there must be a representation that can be understood by at least one other agent.

The new interpretation of the architecture is designed to make use of meta-level information in order to encode control information. Even though the individual agents that comprise a system are somewhat restricted in power, together they have enough power to reason about themselves as a complete system; indeed, we foresee systems that contain agents that are dedicated to meta-level reasoning and to performing meta-level tasks. Such agents would have access to the contents of the blackboard process in the normal way, but would perform tasks at a level higher than most of the other agents in the system. Meta-level agents would also have the ability to bring other agents into existence and to remove them from the system. Because of the possibility that systems communicate in a variety of ways (principally, over a communications network), meta-level agents may also be responsible for initiating and terminating communications channels (this is, it must be admitted, to be examined in more detail).

This recursive structure conveys great power to the new structure. The new architecture is designed in terms of independent processes. Each agent is a process, as is the blackboard. Agents may employ idiosyncratic internal representations: they must agree on the public representations which they use to communicate with each another on the blackboard. In addition to this concurrency, the architecture is also designed to be distributed. It is possible to connect systems using a non-local communications mechanism (indeed, it is possible to distribute the components of a single system so that, for example, the blackboard process resides on a processor different from that on which one or more agents execute) into a network of systems. We have discussed this as well as the possibility of recursively embedding systems in other systems of similar structure. For example, what appears in one system to be a single agent may, in reality, be an entire system composed of agent and blackboard processes.

We first stated our new interpretation of the blackboard architecture in a lecture during February, 1992. It is, however, far older in conception and resulted (in a fashion analogous to our CASSANDRA architecture (Craig, 1989, 1991a)) from dissatisfaction with the conventional (HEARSAY-II based) interpretation of the blackboard metaphor. The concept of embedding—which we find particularly exciting because of its promise and because of its resemblance to many theories in cognitive science (see Johnson-Laird, 1993)—originally occurred to us while working on ways of improving knowledge source power in conventional blackboard systems. Embedding remains one of the problems that we need to address in future work on the new interpretation.

We anticipate constructing a testbed implementation of the new interpretation using EuLisp during the next few months. In the first implementation, we will provide facilities for executing configurations over a network of processors and for instantiating agents and systems so that embedding will be possible. We hope to report on progress in the relatively near future.

REFERENCES

(Brogi, 1991) Brogi, A. and Ciancarini, P., The Concurrent Language, Shared Prolog, *TOPLAS*, Vol. 13, No. 1, pp. 99-123, 1991.

(Craig, 1987) Craig, I.D., *The BB-SR System*, Department of Computer Science, University of Warwick, Research Report No. 94.

(Craig, 1988) Craig, I.D., Blackboard Systems, *Artificial Intelligence Review* Volume 2, pp. 103-118,1988.

(Craig, 1989) Craig, I.D., *The CASSANDRA Architecture*, Ellis Horwood, Chichester, 1989.

(Craig, 1991a) Craig, I.D., *The Formal Specification of Advanced AI Architectures,* Ellis Horwood, Chichester, 1991.

(Craig, 1991b) Craig, I.D., *Rule Interpreters in ELEKTRA*, Department of Computer Science, University of Warwick, Research Report 191b.

(Craig, 1992) Craig, I.D., *Blackboard Systems,* Ablex Pub. Corp., Norwood NJ, 1992.

(Craig, 1993a) Craig, I.D., Formal Techniques in the Development of Blackboard Systems, *International Journal of Pattern Recognition and Artificial Intelligence*, 1993.

(Craig, 1993b) Craig, I.D., A Reflective Production System, *Kybernetes*, to appear.

(De Bosschere, 1993) De Boschere, K., Jacquet, J.-M. and Tarau, P. (eds.), *Proc. ICLP 93 Post Conference Workshop on Blackboard-Based Logic Programming*, Budapest, 1993.

(Engelmore, 1998) Engelmore, R. and Morgan, T., *Blackboard Systems*, Addison-Wesley, Wokingham, 1988.

(Erman, 1975) Erman, L.D, and Lesser, V.R., A Multi-level Organization for Problem Solving Usign Many, Diverse, Cooperating Sources of Knowledge, *Proc. IJCAI-4*, Vol. 2, pp. 483-490, 1975.

(Gelernter, 1985) Gelernter, D., Generative Communication in Linda, *TOPLAS*, Vol. 7, No. 1, pp. 80-112, 1985.

(Gelernter, 1992) Gelernter, D., *Mirror Worlds*, OUP, 1992.

(Hayes-Roth, 1979) Hayes-Roth B. and Hayes-Roth, F., A Cognitive Model of Planning, *Cognitive Science*, Vol. 3, pp. 275-310, 1979.

(Hayes-Roth, 1985) Hayes-Roth, B., A Blackboard Model for Control, *Artificial Intelligence*, Vol. 26, pp. 251-322, 1985.

(Hayes-Roth, 1986) Hayes-Roth, B., Garvey, A., Johnson, M.V. and Hewitt, M., *BB*: A Layered Environment for Reasoning about Action*, Technical Report No. KSL 86-38, Knowledge Systems Laboratory, Stanford University, 1986.

(Johnson-Laird, 1993) Johnson-Laird, P.N., *The Computer and The Mind*, 2nd edition, Fontana Press, London, 1993.

(Newell, 1962) Newell, A., Some problems of the basic organization in problem-solving programs, Yovits, M.C., Jacobi, G.T. and Goldstein, G.D. (eds), Proc. Second Conference on Self-Organizing Systems, pp. 393-423, Spartan Books, 1962.

(Nii, 1986a) Nii, H.P., Blackboard Systems Part One, *AI Magazine*, Vol. 7, No. 2, pp. 38-53, 1986.

(Nii, 1986b) Nii, H.P., Blackboard Systems Part Two, *AI Magazine*, Vol. 7, No. 3, pp. 82-106, 1986.

(Nii, 1986c) Nii, H.P. and Rice, J.P., *CAGE and POLIGON: two Frameworks for Blackboard-Based Concurrent Problem Solving*, Technical Report KSL 86-41, Knowledge Systems Laboratory, Stanford University, 1986.

(Padget, 1993) Padgett, J. *et al.*, *Programming Language Eulisp*, School of Mathematics, University of Bath, 1993.

(Rice, 1986) Rice, J.P., *POLIGON: A System for Parallel Problem Solving*, Technical Report No. KSL 86-19, Knowledge Systems Laboratory, Stanford University,

1986.
(Schwartz, 1993) Schwartz, D.G. and Stirling, L.S., *BackLog: From Blackboard System to Process-Oriented Prolog*, in De Boschere, 1993.
(Velthuijsen, 1992) Velthuijsen, H., *The Nature and Applicability of the Blackboard Architecture*, Ph.D. thesis, Dept. of Computer Science, University of Limburg, Maastricht, The Netherlands, 1992.