UNIVERSITY OF ZIELONA GÓRA
Faculty of Management
and Faculty of Mechanical Engineering
In association with the Design Society

FOURTH INTERNATIONAL SEMINAR AND WORKSHOP
7th - 9th October 2004
Zielona Góra - Rydzyna
POLAND

Engineering Design in Integrated Product Development
Management of Design Complexity

COLLABOLRATIVE CONTROL SYSTEM BASED ON BLACKBOARD ARCHITECTURE APPROACH

M. GIL
Warsaw University of Technology
Department of Machine Design Fundamentals
e-mail: maciej.gil@ipbm.simr.pw.edu.pl

Keywords: expert systems, blackboard architecture, distributed problem solving, agents, knowledge-source, CLIPS,

Abstract: *The presented system is a generic framework for data and knowledge exchange between set of intelligent agents. These intelligent agents run as programs on Windows environment and communicate using DDE, a form of inter process communication based on sending reserved window-events. Moreover, each program is a wrapper for CLIPS expert system shell, and thus each has its own knowledge 'loaded' from CLIPS file and performs its own task in the group. The implementation of DDE enabled intelligent agents to exchange simple facts, rules, and even CLIPS functions at the session run time. One of the Agents is delegated as master and controls the order of execution of the others. The system can be used as a generic blackboard architecture, which can run on Windows platforms and utilizes CLIPS language for implementation of knowledge sources.*

INTRODUCTION

Modern computing and information systems are distributed, large, open, and heterogeneous. Thanks to internet, computers are no longer stand-alone, hermetic systems, but have became integrated both with each other and their users. This trend can be observed on many areas, and among others, touches Computer Aided Design. The increasing diversity and complexity of CAD systems, the market pressure for rapid design and development, goes together with an increasing number and complexity of CAD applications. The nature of CAD is almost each new project or design is unique. Every time the designer, user of a CAD application, copes with new goals, conditions, and constraints. Only one aspect is quasi permanent – the knowledge of the designer, or knowledge embedded in the CAD application. To cope with such applications, computers have to act more as "individuals", rather than just "components".
The designers interact in various ways and at many levels: they observe, model, they request or provide information, they negotiate or discuss, they share their work, they detect conflicts and they try to and resolve them. When the group of designer is large enough, they form and dissolve organizational structures. There are interactive processes among designers- human beings and similar interaction must be mirrored on the computer layer.
To satisfy the above concerns distributed artificial intelligence emerged in late 1970s and since its inception evolved and diversified rapidly. [5], [6], [7], [8], [9], [10].

1. BLACKBOARD APPROACH

Distributed problem solving in artificial intelligence deals with solving a problem in a decentralized environment through cooperation among a set of intelligent entities called knowledge sources or agents. Each agent can run in parallel with others. They can run on a single computer or can be distributed geographically. The key issue in this approach is the control, communication and knowledge sharing among participating intelligent agents, necessary to produce valuable solution. The blackboard model of problem solving is one of the most common approaches. In this section blackboard architecture approach as a model for distributed problem solving is presented. Blackboard-based problem solving is inherited its name from the following metaphor:

"Imagine a group of human or agent specialists seated next to a large blackboard. The specialists are working cooperatively to solve a problem, using the blackboard as the workplace for developing the

solution. Problem solving begins when the problem and initial data are written onto the blackboard. The specialists watch the blackboard, looking for an opportunity to apply their expertise to the developing solution. When a specialist finds sufficient information to make a contribution, he records the contribution on the blackboard. This additional information may enable others to apply their expertise. This process of adding contributions to the blackboard continues until the problem has been solved." [2]. This is quite straightforward metaphor but captures a number of the important features blackboard systems have, which are independence of expertise, diversity in problem-solving techniques, flexible representation of blackboard information, common interaction language, event-based activation, need for control and incremental solution generation. [3], [6], [9].

2. IMPLEMENTATION

The goal of this project was to design and develop generic framework for distributed problem solving, where agents are implemented as CLIPS-based instances working on the Windows platform. Because of the above assumptions, two major problems had to be solved; first- to develop a set of windows applications having the ability to read and run CLIPS programs [1]; second- make the applications interacting dynamically. The following sections describe current solutions.

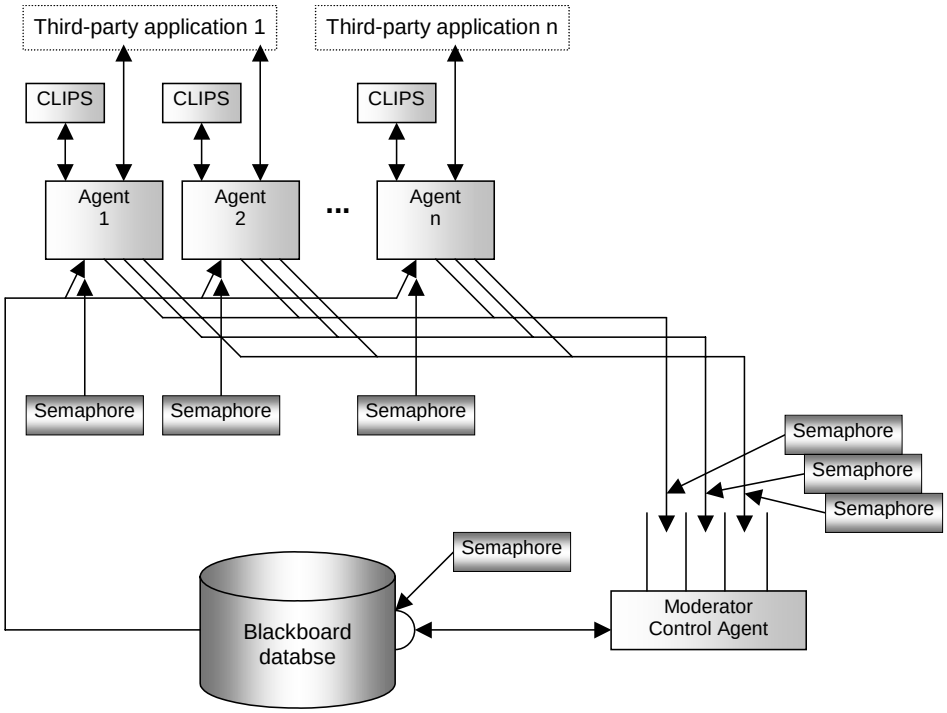

Fig. 1. *The architecture of the blackboard system with agents, control and blackboard implemented as separate programs communicating via DDE windows layer.*

2.1. CLIPS wrapper

CLIPS is expert system tool based on the original OPS5 language developed by Charles Forgy [1], [9]. OPS5 has been widely used for expert system development because it used a very efficient pattern-matching algorithm (the Rete network). Researchers at NASA re-implemented OPS5 in the C language, renaming it CLIPS. Mark Tomlinson re-implemented CLIPS as ActiveX Control, that allows programmers to embed the artificial intelligence engine into C++ and Visual Basic Programs. The ActiveX control has been then utilized in the CLIPS wrapper application developed as a part of the presented system.

2.2. DDE communication

The (DDE) - Windows Dynamic Data Exchange mechanism is actually a protocol, a set of guidelines that enables DDE-compatible Windows applications to share data easily with other DDE compatible applications. DDE one can use to perform one-way data transfers or to perform ongoing conversation. In real-time conversations, applications send updates to one another as new data becomes available. Figure 1 shows detailed architecture of the developed system and DDE communications channels between the components.

3. ARCHITECTURE

The system is divided into three components. First is the Blackboard Table program, which serves as a central storage area for other components. Second – is the group of intelligent agents programs [4]. These are instances of one generic CLIPS wrapper program, each of which has its own logic loaded from the particular clips source file. Third is a application called 'Mediator' – this one plays the role of control mechanism responsible for the strategy of the execution of other Intelligent Agents programs (Fig.1.). All components have been implemented in MS Visual Basic and are described in detail in the following sections.

3.1. Intelligent Agents

In general, an intelligent agent for this system can be any Windows application, which is compatible with DDE, i.e. has an interface for communications channel between running applications. In this prototype application, the dedicated programs were developed for this purpose. The core of them was CLIPS ActiveX control – Visual Basic component that played as a local expert system shell using rule-based knowledge representation scheme. The Agent program has a special interface enabling it to receive messages from other parallel running programs, components of the system, especially from the blackboard program. Receiving such messages, the Agent can process it, reason and perform computations (or communicate with third-party, external applications). The output of this computation then can be sent back to the mediator program, which publishes it in the certain form on the blackboard. The change on the blackboard database can activate other agents and so on.

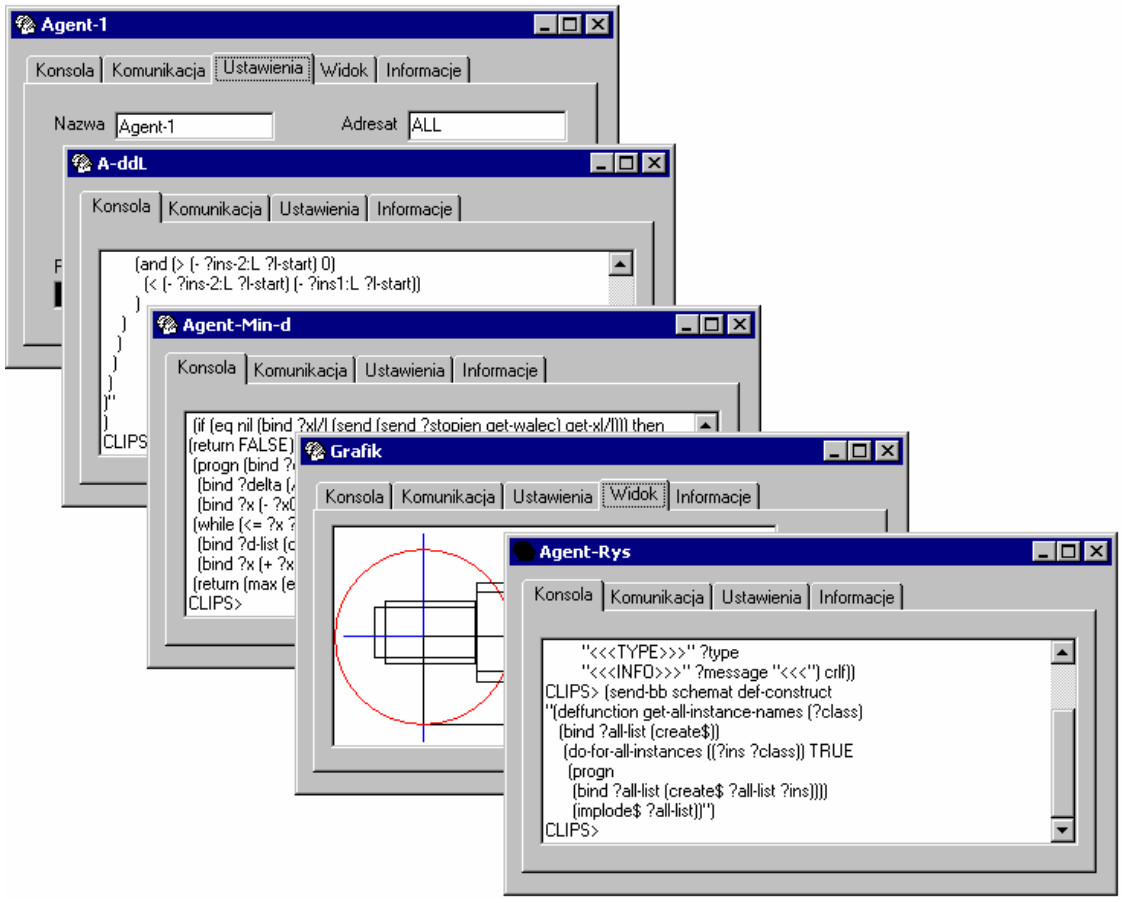

Fig. 2. *A set of sample intelligent agents working around the blackboard solving a sample problem of mechanical shaft design*

Currently, each of the agents works as a separate instance (run-time) of a CLIPS wrapper program, called Clips-Agent. To load and activate CLIPS based intelligent agent this clips-agent.exe should be executed. Then the initialization window is being opened, where the user is asked to provide this Agent (this instance) name, the knowledge base i.e. file that contains CLIPS rules and the Group name. Group is useful to categorize messages sent by Blackboard program.

The source program structure (on the Visual Basic level) is design in the way the program can be easily redesigned and recompiled – if there will be any customization requirement. Some extensions to the basic Clips-Agent are developed, e.g. one of the agents was deployed with GUI, and was able to show all mechanical parts currently posted on the blackboard on its graphic widget. There is also a possibility to create an interface to third-party or legacy applications, which are external to this system. Figure 2 shows sample set of intelligent agents programs working as a clone of the original Clips-Agent application or the programming extension of it.

3.2. Control Agent Program

Control agent is the part of the system, which behavior has a strong influence to the overall result. It has been also implemented using Visual Basic programming environment and deploying CLIPS ActiveX component. It runs as a separate program and communicates with the other agents and the blackboard program using DDE facilities. The control can be loaded by calling mediator.exe in Windows. This program has to be loaded before any of other agents. This program, after successful start and loading CLIPS knowledge base from the associated file, opens three separate DDE channels. DDE facility has been used for setting up a communication link from the intelligent agents to the control. The control creates incoming queues to receive messages from the agents. There are three different channels to support better manageability: message channel, command channel and signal channel.

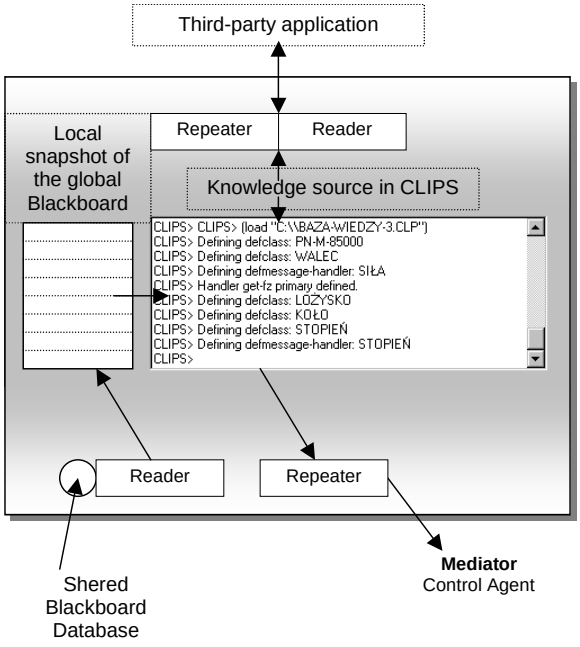

Fig. 3. *The structure of the Intelligent Agent developed in Visual Basic and utilizing CLIPS ActiveX component for wrapping CLIPS knowledge bases.*

The message channel is used to read all messages coming from agents, which are pertaining to the problem being actually solved. The command channel is used for system administration purposes, like e.g. starting, suspending or retracting an agent. Signal channel is used for propagating predefined events, which can be send or broadcasted to the agents. All channels are associated with fifo queues and handled by three lists in VB program (see figure 3.). Currently the control mechanism requires human being to reason and act as a control of the blackboard, thus it has GUI interface (see figure 4.) showing the excel-like data grid to make this task convenient. There is also a possibility to load knowledge base coded in CLIPS format, but there is still lack of success to make it running autonomously in unattended way.

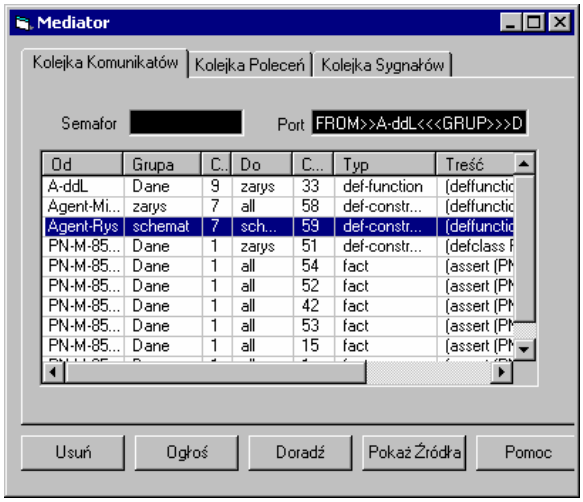

Fig. 4. *The GUI of the Mediator Control Agent program.*

The user interface can display not only message content but also other data which can support to organize the blackboard, i.e. who is the sender of the message, who are the addressee, group of addressees or maybe all agents, what was the time (cycle number) the message was sent, what is the group name of the addressee, message structure, i.e. message content is a valid CLIPS constructs like a fact, or set of facts, a rule, an object or a function..

3.3. Blackboard Program

The blackboard is a shared memory that holds information and knowledge that intelligent agents use. The blackboard is organized by the control component. The blackboard holds the following properties: its name, the name of the level (or space), its status, the cycle counter, and the database sharing area containing rules, facts and commands that needs to be broadcast to the intelligent agents (see figure 5.)

Because there is a lack of shared memory facility in Windows operating system (which is known and useful feature of all Unix platforms) the system

needs to hold one main blackboard for the control and build its local copy of the blackboard for each intelligent agent. Nevertheless this option is useful and prepared for future network extensions. If intelligent agents were distributed on different computer systems across the network, utilizing NetDDE protocol, then a copy of the main blackboard needs to be created anyway and maintained in each computer system.

When more than one agent program wants to send a DDE message to control mechanism, the dead lock problem occurs. Both applications want to use the same channel and both hang their execution. To overcome this the mediator program is equipped with the special objects which I call semaphores – as they are resembles the semaphores feature known in Unix world. Semaphores in this application are the objects having text property. In a moment when one agent application has an open channel to mediator program – this property is set to not NULL. This can be read by other agents and suspend its communication until the first one will finish. When conversation is finished, the semaphore property is set to NULL giving green light for others. All messages received from agents are immediately processed and published on the blackboard.

DDE protocol allows sending only plain text messages, thus sequential format of the message is chosen, where each message is divided in the sections containing information about origin, destination and its content. The messages published on the blackboard and broadcasted to agents has the following format

```
(printout vb
  (str-cat "FROM>>" <source-of-information-name>
  "<<<GRUP>>>" <group-name>
  "<<<DEST>>>" <addressee, agent, group or all>
  "<<<TYPE>>>" <type-of-information>
  "<<<INFO>>>" <content> "<<<"
) crlf )
```

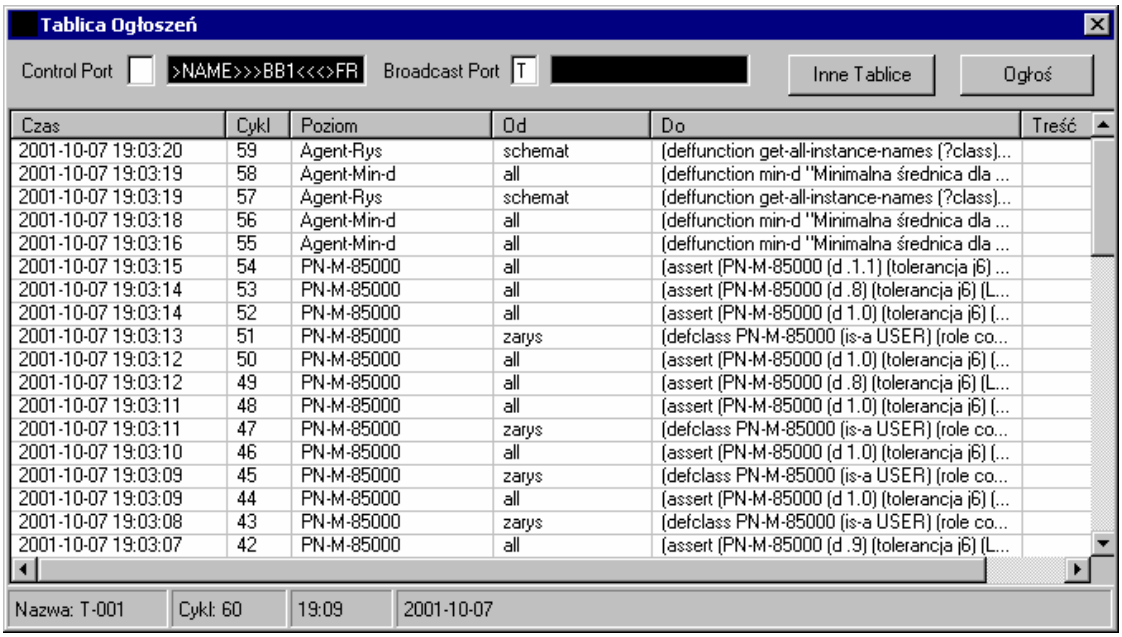

Fig. 5. *The GUI of the Blackboard. Dynamic report shows the history of activities on the blackboard database program.*

4. CONCLUSION AND FUTURE EXTENSIONS

This article presents generic framework based on blackboard architecture approach, where intelligent agents are implemented as a CLIPS wrappers in Visual Basic and can exchange information in a dynamic environment using DDE communication protocol [4].

Using this system any CLIPS- compatible fact, rule or command can be maintained within one intelligent agent and if needed- broadcasted to the other agents. There is one specific agent program playing the role of control structure- Its logic is implemented as well in CLIPS and it utilizes the certain strategy of execution, in order to successfully resolve problem set.

As it was noted in the introduction, dynamic knowledge exchange would be an important feature for any distributed application in which cooperative

problem solving sessions can be initiated where each intelligent agent can share its problem relevant knowledge with other intelligent agents to resolve the problem.

* * *

The proposed system can be expanded to accommodate hierarchy of blackboards. In such case one control program is required for each additional blackboard and DDE communication queues need to be added to additional blackboards. The system was tested on a single computer. The plans are to implement it to use NetDDE, the extension of the DDE protocol enabling the dynamic inter process communications over the network. Hence, intelligent agents cold be geographically distributed but able to share the data dynamically via the system main blackboard. There are also plans to use XML format in the future to transfer facts, roles and functions.

References

[1] Clips Manual. NASA Software technology manuals, CLIPS 6.10, manual, version 6.0; Athens; 1993

[2] I. Craig, *Blackboard Systems*, Ablex Publishing Corporation, Norwood, New Jersey, 1995

[3] R. Engelmore, A. Morgan (Ed.), *Blackboard systems*. Addison-Wesley, New York, 1988

[4] M. Gil, *Proces projektowania w budowie maszyn z zastosowaniem narzędzi komputerowych do integracji jego elementów*, WPW, Warszawa 2001

[5] B. Hayes-Roth, *A Blackboard Architecture for Control*, an International Journal of Artificial Intelligence; Elsevier, 1985, pp. 251-321

[6] C.S. Krishnamoorthy; S. Rajeev , Artificial Intelligence and Expert Systems for Engineers, CRC Press LLC, 1996.

[7] S.E. Lander, *Issues in Multiagent Design Systems*, IEEE Expert, Vol. 12, No. 2, March/April 1997

[8] H. Penny Nii, *Blackboard Systems at the Architecture Level*, Expert Systems With Applications Vol 7; Pergamon; 1994; pp: 43-54

[9] G. Weiss [Ed.], *Multiagent Systems – A Modern Approach to Distributed Modern Approach to Artificial Intelligence*, The MIT Press, Cambridge, MA, London, England, 1999

[10] S. J. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach*, Prentice Hall, Englewood Cliffs, New Jersey, 1995