Monitoring and Diagnostics with Intelligent Agents Using Fuzzy Logic

Arnulfo Alanis Garza, Juan José Serrano, Rafael Ors Carot, Karim Ramirez ,José Mario García-Valdez, Hector Arias, and Jose Soria

Abstract- Intelligent Agents have originated a lot of discussion about what they are, and how they are different from general programs. We describe in this paper a new paradigm for intelligent agents. This paradigm helped us deal with failures in an independent and efficient way. We proposed three types of agents to treat the system in a hierarchical way. A new way to visualize fault tolerant systems (FTS) is proposed, with the incorporation of intelligent agents, as they grow and specialize create the Multi-Agent System (MAS). The MAS contains a diversified range of agents, which depending on the perspective will be specialized or evolutionary (from our initially proposal) they will be specialized for the detection and possible solution of errors that appear in an FTS). The initial structure of the agent is proposed in [1] and named reflected agent with an internal state and in the Method MeCSMA [2]. The present work is based on the idea that with the help of the paradigm of intelligent agents, we may be able to handle fault tolerant systems, in the modality of embedded systems. The idea is to detect errors and to try to correct the failures that could happen in industrial control by monitoring and diagnosis.

***Index Terms*— Intelligent Agents, Fuzzy Logic, Monitoring, Diagnostics.**

I. Introduction

At the present time the approach by means of agents for real applications, has worked with movable agents, which work at the level of the client-server architecture. However, in systems where the requirements are higher, as in the field of the architecture of embedded industrial systems, the idea is to innovate in this area by working with the paradigm of intelligent agents. Also, it is a good idea in embedded fault tolerant systems, where it is a new and good strategy for the detection and resolution of errors.

The main goals of the present research work were the following:

1 To create a new visualization tool of the application of the intelligent agents, in the fault tolerant systems in embedded systems.

To create a model, that helps the programmers to create profiles in the embedded circuits, according to utility, by means of, Intelligent Agents

The reflected agent with an internal state, of [3] Figure 1, sets out the general structure of the recovery Intelligent Agent for Fault tolerant Systems in Distributed Systems, whit three types of intention agents.

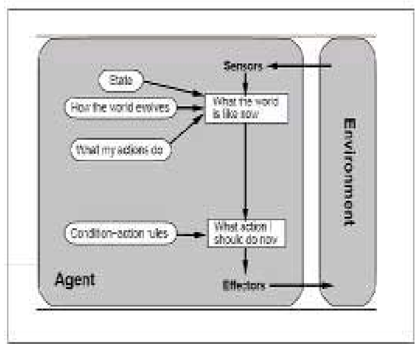

Figure 1 Agent with an internal state.

A. Agents

Let's first deal with the notion of intelligent agents. These are generally defined as "software entities" which assist their users and act on their behalf. Agents make your life easier, save you time, and simplify the growing complexity of the world, acting like a personal secretary, assistant, or personal advisor, who learns what you like and can anticipate what you want or need. The principle of such intelligence is practically the same of human intelligence. Through a relation of collaborationinteraction with its user, the agent is able to learn from himself, from the external world and even from other agents, and consequently act autonomously from the user,

A.Alanis is with the Tijuana institute of technology, Division Graduate and studies Reseach, Calzada Tecnologico, S/N, Tijuana, BC 22379 Mexico :664-682-72-79; e-mail: alanis@ tectijuana.mx).
J. Jose Serrano is with the Universidad Politécnica de Valencia (España), D. Inf. de Sistemas y Computadoras, Camí de Vera, s/n, 46022 VALÈNCIA, ESPAÑA, 00+34 96387, email:jserrano@disca.upv.es.
R.Ors Carot is with the Universidad Politécnica de Valencia (España), D. Inf. de Sistemas y Computadoras, Camí de Vera, s/n, 46022 VALÈNCIA, ESPAÑA, 00+34 96387, email:rors@disca.upv.es.
J. Mario Garcia, Karim Ramirez, Hector Arias and Jose Soria are with the Tijuana institute of technology, Division Graduate and studies Reseach, Calzada Tecnologico, S/N, Tijuana, BC 22379 Mexico :664-682-72-79; e-mail: mario@ tectijuana.mx.

adapt itself to the multiplicity of experiences and change its behavior according to them. The possibilities offered for humans, in a world whose complexity is growing exponentially, are enormous [1], [4], [5], [6].

B. *Multi-Agent Systems*

In artificial intelligence research, agent-based systems technology has been hailed as a new paradigm for conceptualizing, designing, and implementing software systems. Agents are sophisticated computer programs that act autonomously on behalf of their users, across open and distributed environments, to solve a growing number of complex problems. Increasingly, however, applications require multiple agents that can work together. A multi-agent system (MAS) is a loosely coupled network of software agents that interact to solve problems that are beyond the individual capacities or knowledge of each problem solver.

Advantages of a Multi-Agent Approach An MAS has the following advantages over a single agent or centralized approach:

- An MAS distributes computational resources and capabilities across a network of interconnected agents. Whereas a centralized system may be plagued by resource limitations, performance bottlenecks, or critical failures, an MAS is decentralized and thus does not suffer from the "single point of failure" problem associated with centralized systems.
- An MAS allows for the interconnection and interoperation of multiple existing legacy systems. By building an agent wrapper around such systems, they can be inporporated into an agent society.
- An MAS models problems in terms of autonomous interacting component-agents, which is proving to be a more natural way of representing task allocation, team planning, user preferences, open environments, and so on.
- An MAS efficiently retrieves, filters, and globally coordinates information from sources that are spatially distributed.
- An MAS provides solutions in situations where expertise is spatially and temporally distributed.
- An MAS enhances overall system performance, specifically along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility, and reuse [3].

II. Distributed Artificial Intelligence

Distributed Artificial Intelligence (DAI) systems can be defined as cooperative systems where a set of agents act together to solve a given problem. These agents are often heterogeneous (e.g., in Decision Support System, the interaction takes place between a human and an artificial problem solver).

Its metaphor of intelligence is based upon social behavior (as opposed to the metaphor of individual human behavior in classical AI) and its emphasis is on actions and interactions, complementing knowledge representation and inference methods in classical AI.

This approach is well suited to face and solve large and complex problems, characterized by physically distributed reasoning, knowledge and data managing. In DAI, there is no universal definition of agent, but Ferber's definition is quite appropriate for drawing a clear image of an agent: "An agent is a real or virtual entity which is emerged in an environment where it can take some actions, which is able to perceive and represent partially this environment, which is able to communicate with the other agents and which possesses an autonomous behavior that is a consequence of its observations, its knowledge and its interactions with the other agents".

DAI systems are based on different technologies like, e.g., distributed expert systems, planning systems or blackboard systems. What is now new in the DAI community is the need for methodology for helping in the development and the maintenance of DAI systems. Part of the solution relies on the use of more abstract formalisms for representing essential DAI properties (in fact, in the software engineering community, the same problem led to the definition of specification languages) [7] [8].

III. Proposed Method

Be a Distributed System (mainly applied to the industrial control), which this made up of a set of Nodes, where each one of them can be constituted by several Devices.

On these Nodes a set of Tasks, ordered is executed all of them to take I finish the functionality of the system. In order to identify this Distributed System the following definitions set out:

Definition 1: = is N {Nor}, the set of the Nodes of the system, being n is the number of units that integrate it.

Definition 2: Be [Di, z], the set of devices that contains Node i. Where z can take value 1, if it is wanted to see the Node like only device, or greater than 1 if it is desired to do visible some of the elements that integrate it.

Definition 3: = is T {T j}, the set of tasks that are executed in the system, being t the number of tasks that integrate the system.

Definition 4: A System Distributed like dupla is defined: SD = (N, T) Once characterized what a Distributed System could be denominated Basic (without no characteristic of Tolerance to Failures), is going away to come to the incorporation on he himself from the paradigm of Intelligent Agents with the purpose of equipping it with a layer with Tolerance to Failures. The Fault tolerant Agents will define themselves now that worked in the SD

Definition 5: An Agent is ANi to whom Agent denominated itself Node, whose mission is the related one to the tolerance to failures at level of the Node Nor. An Agent exists therefore Node, by each Node of the System (biyectiva application).

Definition 6: An Agent is ATj to whom Agent denominated itself Task, whose mission is the related one to the tolerance to failures at level of the Tj Task. An Agent exists therefore Task by each task of the system.

Definition 7: An Agent is AS to whom Agent denominated itself to him System, whose mission is the related one to the tolerance to failures system level.

Definition 8: AN= {ANi} the set of all the Agents Node.

Definition 9: AT= {ATj} the set of all the Agents Task.

Definition 10: A System is SMATF Fault tolerant Multi-Agent, formed by tripla of ANi, ATj, AS.

That is to say, SMATF = <Ni, AT j, AS> with it a Distributed System Fault tolerant SDTF is defined as:

Definition 11: A Distributed System Fault tolerant SDTF like dupla SDTF is defined SDTF =<SD, SMATF > Next goes to describe with greater detail each one of the Agents who compose the SMATF [9], [10].

IV. Implementation of the Approach

We have implemented the intelligent agents using the MATLAB programming language. The reason for using this programming language is that it is very easy to develop prototypes. At the beginning the implementation was done with traditional Boolean logic. In other words, the if-then rules of the agent were considered as categorical (as in an ideal situation with no uncertainty). We show in Figure 6 the implementation in MATLAB of one of the intelligent agents with categorical if-then rules. In a second phase, we consider using fuzzy logic to model uncertainty in the decision process. In this case, the knowledge base consists of fuzzy rules. We show in Figures 7 and 8 the implementation in MATLAB of the fuzzy rules of an intelligent agent.

```
% AGENTE SISTEMA

 %Detection
        if AS.Phase.Detection && AN(i).S..Phase. operation free of
          errorLocalitation
        end
        if AS.Phase.Detection &&  ATjS.Inexecution
          free of error
        end
        if AS.Phase..Detection &&  ATj.Down
          toterminatethedeviceofthenode && AS.Phase.Reconfiguration
        end
        if AS.Phase..Detection &&  ATjS.State.Software.Error
          ASNi.Functiondeteted Error.  ASTj.lengthy.execution &&
          AS.Phase.Location
        if AS.Phase..Detection &&  ATjS.State.Hardware Error
          ASNi.Functiondeteted Error.
          ASTj.lengthy.execution &&
          AS.Phase.Location
        end
        if AS.Phase..Detection &&  ATj.Degraded
          toterminatethedeviceofthenode && AS.Phase.Reconfiguration
        end
%Location
        if AS.Phase.Location &&ANiS.underrecovery
          AS.Phase.Isolation
        end
%Isolation
        if AS.Phae.Isolation ANiS.Correct.State
        AS.Phase Recovery
        end
        if AS.Phae.Isolation ANiS.Down.State
        AS.Phase Reconfiguration
        end
        if AS.Phae.Isolation ANiS.Degraded.State
        AS.Phase Reconfiguration
        end
% Reconfiguration
        if AS.Phase.Reconfiguration && ANi.State.Down o Degraded
          AS.Phase.Recovey
        end
% Recovery
        if AS.Fase.Recovery && ASTj.EjecucionReconRecu
          AS.Phase.Detection
        end
```

Fig. 6 Knowledge base of the intelligent agent.

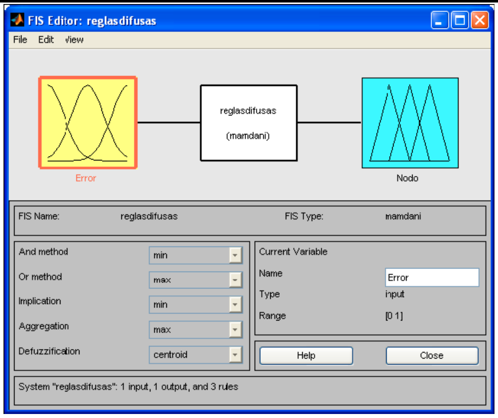

Fig. 7 Fuzzy system structure of the intelligent agent.

```
a=newfis('reglasdifusas');
a.input(1).name='error';
a.input(1).range=[0 1];
a.input(1).mf(1).name='bajo';
a.input(1).mf(1).type='trapmf';
a.input(1).mf(1).params=[-0.25 0 0.2
0.5];
a.input(1).mf(2).name='medio';
a.input(1).mf(2).type='trimf';
a.input(1).mf(2).params=[0.2 0.5 0.8];
a.input(1).mf(3).name='alto';
a.input(1).mf(3).type='trapmf';
a.input(1).mf(3).params=[0.5 0.8 1
1.12];

a.output(1).name='nodo';
a.output(1).range=[0 1];
a.output(1).mf(1).name='Baja'
a.output(1).mf(1).type='trapmf';
a.output(1).mf(1).params=[-0.25 0 0.1
0.4];
a.output(1).mf(2).name='Degradado';
a.output(1).mf(2).type='trimf';
a.output(1).mf(2).params=[0.2 0.5 0.8];
a.output(1).mf(3).name='OK';
a.output(1).mf(3).type='trapmf';
a.output(1).mf(3).params=[0.6 0.9 1
1.25];

a.rule(1).antecedent=[1];
a.rule(1).consequent=[3];
a.rule(1).weight=1;
a.rule(1).connection=2;

a.rule(2).antecedent=[2];
a.rule(2).consequent=[2];
a.rule(2).weight=1;
a.rule(2).connection=1;

a.rule(3).antecedent=[3];
a.rule(3).consequent=[1];
a.rule(3).weight=1;
a.rule(3).connection=2

fuzzy(a)
```

Fig. 8 Fuzzy rule base implementation of the agent.

V. Simulation Results

We show in this section the simulation results of the intelligent agent that was implemented in MATLAB. We show results of two different fuzzy models used in the intelligent agents, one of Mamdani form and the other in Sugeno style. First, we show the results of the Mamdani fuzzy model in Figures 9, 10, 11, 12, and 13. In Figure 9 we show the use of the rule viewer of the MATLAB fuzzy logic toolbox to test the fuzzy rules. In Figures 10 and 11 we show the membership functions used in the fuzzy rules. In Figure 12 we show the non-linear surface of the fuzzy system. In Fig. 13 we show the fuzzy rules for the error

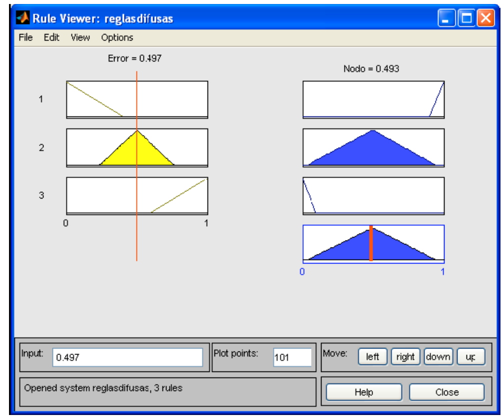

Fig. 9 Rule viewer to test the fuzzy system.

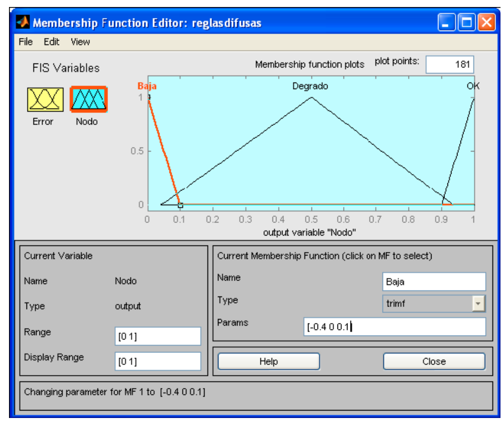

Fig. 10 Output membership functions.

Fig. 11 Input membership functions.

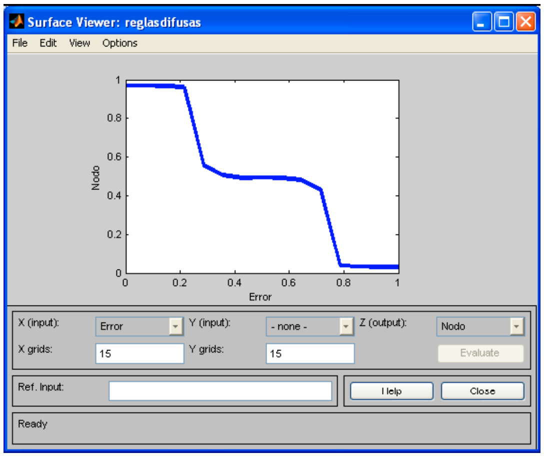

Fig. 12 Non-linear surface of the fuzzy system.

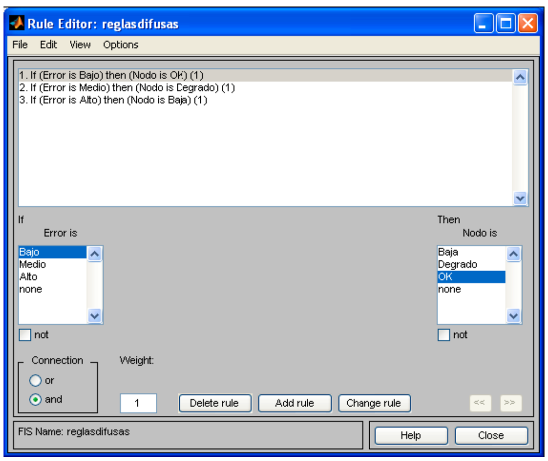

Fig. 13 Fuzzy rules of the system.

Now we show the results of the Sugeno fuzzy model in Figures 14, 15, 16, 17, 18 and 19. In Figure 14 we show the structure of the Sugeno fuzzy model. In Figure 15 we show the use of the rule viewer of the MATLAB fuzzy logic toolbox to test the fuzzy rules. In Figure 16 we show the non-linear surface of the fuzzy system. In Figure 17 we show the membership functions used in the fuzzy rules. In Fig. 18 we show the constant output values of the Sugeno model. In Figure 19 we show the fuzzy rules for the error in monitoring the process. The simulation results of the Sugeno fuzzy model are better than the results of the Mamdani model and it is selected as the better option for this problem of monitoring and diagnostics

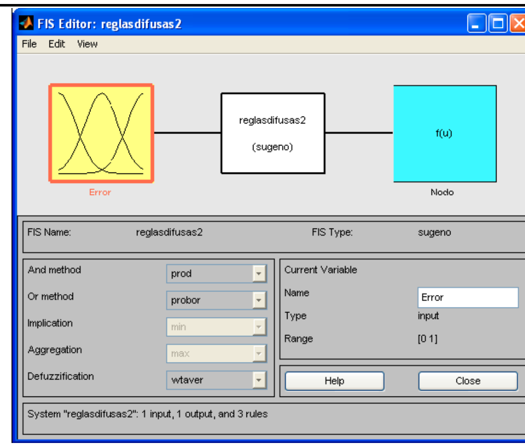

Fig. 14 General structure of the Sugeno model.

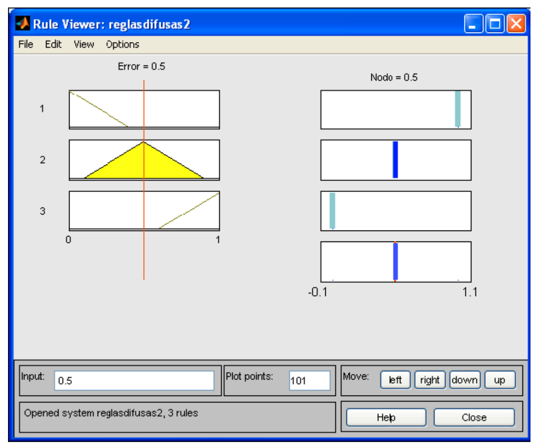

Fig. 15 Use of the rule viewer to test the system.

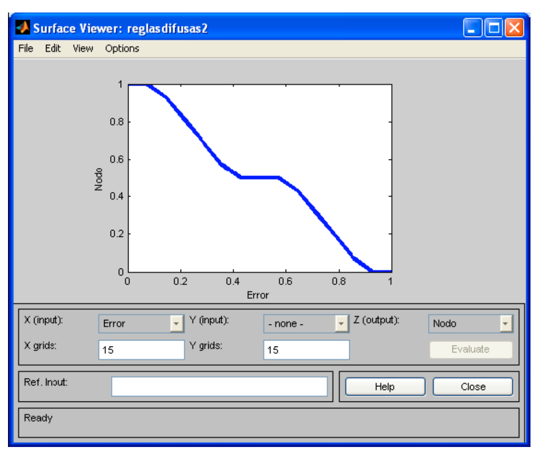

Fig. 16 Non-linear surface of the fuzzy system.

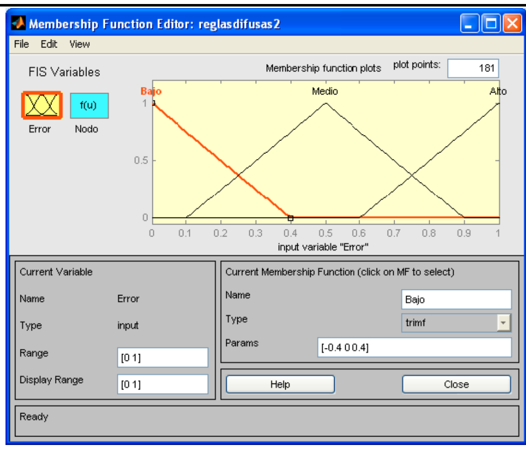

Fig. 17 Input membership functions of the system.

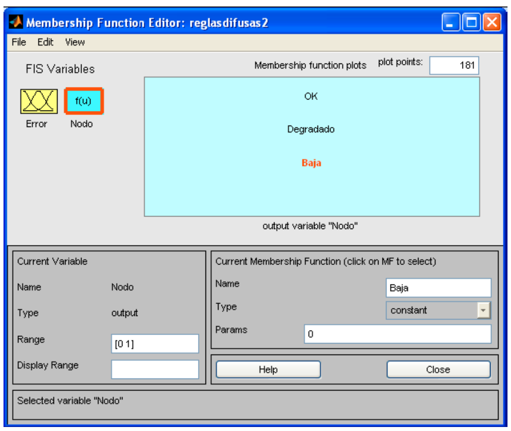

Fig. 18 Constant output values of the rules.

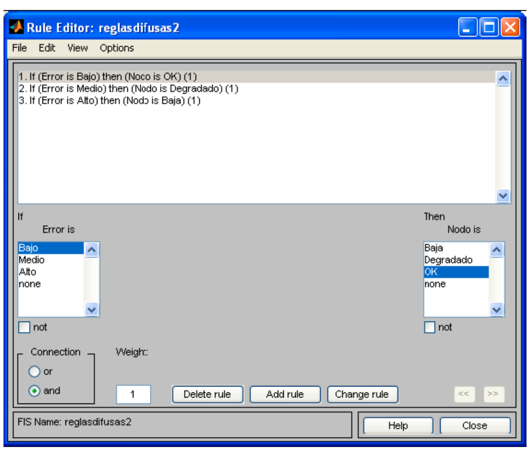

Fig. 19 Rules of the fuzzy Sugeno model.

VI. Conclusions

We described in this paper our approach for building multi-agent systems for achieving fault tolerant control system in industry. The use of the paradigm of intelligent agents has enabled the profile generation of each of the possible failures in an embedded industrial system. In our approach, each of the intelligent agents is able to deal with a failure and stabilize the system in an independent way, so that the system has a behavior that is transparent for the application as well as for the user. An intelligent agent with fuzzy logic is able to monitor and make diagnosis of possible problems in the process.

References

[1] Stuart Russell and Peter Norvig, ArtificialIntelligence to Modern Approach, Prentice artificial Hall series in intelligence, Chapter Intelligent Agent, pages. 31-52

[2] A. Alanis, Architectures for Systems Multi-Agentes, (Master Degree thesis in computer sciences), Tijuana Institute of Technology, November, 1996.

[3] Michael J. Woodridge, Nicholas R. Jennings.(Eds.), Intelligence Agents, Artificial Lecture Notesin 890 Subseries of Lectures Notes in ComputerScience, Amsterdam, Ecai-94 Workshop on Agent Theories, Architectures, and languages, The Netherland, August 1994 Proceedings, ed. Springer-Verlag, págs. 2-21.

[4] P.R. Cohen ET ,An Open Agent Architecture,working Notes of the AAAI Spring symp.: SoftwareAgent, AAAI Press, Cambridge, Mass., 1994 págs.1-8.

[5] Bratko I. Prolog for Programming Artificial Intelligence, Reading, Ma. Addison-Wesley, 1986.

[6] Or Etzioni, N. Lesh, and R. Segal,Bulding forSoftbots UNIX? (preliminary report). Tech. Report93-09-01. Univ. of Washington, Seattle, 1993

[7] Elaine Rich, Kevin Knight, Artificial intelligence, Second Edition, Ed. Mc Graw-Hill,págs. 476-478.

[8] Elaine Rich, Kevin Knight, Artificial intelligence, Second.Edition, Ed. Mc Graw-Hill,págs. 478-479.

[9] Arnulfo Alanis Garza , Juan José Serrano, Rafael Ors Carot, José Mario García Valdez, jornal Engineering Letters. special issue, "Hybrid Intelligent System using Neural Networks, Fuzzy Logic, and Genetic Algotrithms.

[10] Arnulfo Alanis Garza , Juan José Serrano, Rafael Ors Carot, José Mario García Valdez, Hybrid Intelligent Systems: Analysis and Design (Studies in Fuzziness and Soft Computing), springer 2007, by Oscar Castillo (Editor), Patricia Melin (Editor), Janusz Kacprzyk (Editor), Witold Pedrycz (Editor).

Arnulfo Alanis Garza, Bachelor in Computer Engineering Systems from Technological Institute of San Luis Potosi. Candidate to the Ph.d. Degree in Computer Science from Polytechnical University of Valencia,. Full time Researcher of Tijuana Institute of Technology Since 1995. Current areas of interest include: Intelligent Agents, Expert System, Robotics and Networks, Fault Tolerance System. Published more 20 papers in conference proceedings and journals.

Juan Jose Serrano, Bachelor in Industrial Engineering from Polytechinical University of Valencia.. Phd Degree in Computer Science form Polytechinical University of Valencia, Full time Researcher of Polytechinical University of Valencia since 1999. Current areas of interest include: Intelligent Fault Tolerance System, Real-time systems, microcontroller.Published more 40 papers in conference proceedings and journals.

.

Rafael Ors Carot, Bachelor in Industrial Engineering from Polytechinical University of Valencia.. Phd Degree in Computer Science form Polytechinical University of Valencia. Full time Researcher of Polytechinical University of Valencia since 1999. Current areas of interest include: Intelligent Fault Tolerance System, Real time systems, micro-controller. Published more 40 papers in conference proceedings and journals.

Karim Ramirez is a student of the Graduate Program in Computer Science of Tijuana Institute of Technology. Current research interests are intelligent agents and fuzzy logic.

Jose Mario Garcia Valdez, Bachelor in Computer Engineering System from Technological Institute of Tijuana. Full time Researcher of Tijuana Institute of Technology Since 1994. Current areas of interest include: data base, objects of learning. Published more 10 papers in conference proceedings and journals.

Hector Arias is a Professor with the Division of Graduate Studies and Research in Tijuana Institute of Technology, Mexico. He has published 4 Journal papers and 5 Conference Proceedings papers. His current research interests are Robotics, Automation and Control. He is member of the Hispanic American Fuzzy Systems Association.

Jose Soria is a Professor with the Division of Graduate Studies and Research in Tijuana Institute of Technology, Mexico. He has published 4 Journal papers and 5 Conference Proceedings papers. His current research interests are Theory of Computing and Quantum Computing. He is member of the Hispanic American Fuzzy Systems Association.