A MAS IMPLEMENTATION FOR SYSTEM IDENTIFICATION AND PROCESS CONTROL

E. J. González, A. Hamilton, L. Moreno, R. L. Marichal, G. N. Marichal, and J. Toledo

ABSTRACT

In this paper, a MAS for system identification and process control is presented. In particular, this MAS implements a self-tuning regulator (STR) scheme. It has adopted FIPA specifications because they have become a stronger standard in MAS development and they involve not only agent language specifications but also agent management and conversations. In this work, an Ontology Agent (OA) is included, using DAML+OIL as ontology language. The obtained results validate this approach in the implementation of well-known algorithms for control process.

KeyWords: Multiagent systems, identification, process control.

I. INTRODUCTION

Agents and Multi-Agent Systems [1,2] (MAS) are part of a new programming paradigm. They have been successfully used in a wide range of applications such as robotics, e-commerce, agent-assisted user training, military transport or health-care. However, why using agents? This is not a trivial question, as there is not a uniform and widely accepted definition of agent. Examples of agent definitions are:

Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed. (Maes [3])

Intelligent agents are software entities that carry out some set of operations on behalf of a user or another program with some degree of independence or autonomy, and in so doing, employ some knowledge or representation of the user’s goals or desires. (IBM [4])

Autonomous agents are systems capable of autonomous, purposeful action in the real world. (Brustoloni [5])

As can be seen, agents are often characterized by describing their features (long-lived, autonomous, reactive, proactive, collaborative, ability to perform in a dynamic and unpredictable environment). Users can delegate to agents tasks that are designed to be carried without human beings intervention, for instance, as resource managers or personal assistants that learn from its user. Nevertheless, in most of applications, a standalone agent is not sufficient for carrying out the task: agents are forced to interact with other agents, forming a multi-agent system (MAS). Due to their capacity of flexible autonomous action, MAS can treat with highly dynamic or uncertain environments. On the other hand, MAS can effectively manage situations where distributed systems are needed: the problem being solved is itself distributed, data are geographically distributed, managed by different control systems and/or difficult to share, systems with many components and huge content. In this sense, it is not exaggerated to say that MAS have become a key tool for AI researchers and developers.

In this context, having a shared ontology for MAS inter-agent communication is critical to successful communication because a shared ontology provides the common format in which express data and knowledge. An ontology is usually defined as a set of classes, relations, functions, etc. that represents knowledge of a particular domain that provides the common format in which express data and knowledge [6]. There are several ontology languages. However, from their experience, the authors prefer those languages known as “markup languages”, whose last generation (RDF, DAML+OIL [7] and OWL) gives computers an extra small degree of autonomy that can help them to do
more useful work for people. Systems may be able to provide all sorts of additional services and responses beyond the requirements of the standard but a certain basic set of conclusions will always be required.

Nevertheless, despite the described advantages, the application of the agent technology and ontologies to the process automation has not been so frequent. It can be mentioned only a few approaches to the application of MAS technology to process automation. In these cases, agents are an integration mechanism among systems designed separately (for example the ARCHON [8] architecture) or, like APACS [9], agents are designed as supervisors of the process. Some reasons for this lack of application can be found in [10]: automation applications need real-time requirements that are out of currently agent technology reach, difficult agent treatment about interrelations and lack of parallelism. Moreover, ontologies have even been less used in this discipline. There is no doubt that MAS and ontologies are fields to explore in process automation.

The main aim of this paper is not to justify the use of MAS and ontologies in a strict way but to present its application to a control problem. In this context, a MAS (called MASCONTROL) for identification and control of processes is presented in this paper. In particular this MAS implements a self-tuning regulator (STR) scheme, so it is important to remark that this paper does not offer a new general control algorithm but a new approach for its development. That is its main contribution: showing the potential that a controller, through the use of MAS ontologies, can control systems in an autonomous way, using actions whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the control.

In a STR scheme, the problem can be treated properly by a MAS due to two reasons. First, there are several different modules in the STR scheme such as identification, estimation of controller parameters and data acquisition. Secondly, most of these modules should work simultaneously, so that each module can be assigned to an agent.

II. BRIEF DESCRIPTION OF FIPA PLATFORM, DAML+OIL LANGUAGE AND EVENET2000 TOOL

2.1 FIPA platform

After the developers have decided to use MAS to control the system, a carefully study about MAS architectures should be done. In this aspect, the authors strongly recommend researchers to select a standard for this aspect. In this way, their agents could widely interact with other systems. In this sense, there are two main standards about MAS architecture: FIPA [11] (adopted in important projects like FACTS [12], Cameleon [13] and MONADS [14]) and OMG MASIF [15]. The authors have selected FIPA standard because these specifications have become a strong standard in MAS development and it involves not only agent management, but also language specifications, conversations, personal assistant, etc. - OMG MASIF does not define any standard about communication among the agents.

FIPA (Foundation for Intelligent Physical Agents) is an organisation whose purpose is to promote the development of generic agent specifications. Its agent management reference model provides the normative framework where FIPA agents exist and operate. The Directory Facilitator (DF) provides yellow pages services to agents that query it to find out services offered by other agents. The Agent Management System (AMS) offers white pages services and it maintains a directory, which contains transport addresses for agents registered in the Agent Platform (AP). Finally, the Message Transport Service (MTS) is the default communication method among agents on different APs (Fig. 1). A more detailed description of FIPA standard can be found in [11].

As stated above, there are many tools for the design and implementation of MAS. After a carefully study, the authors have chosen FIPA-OS as development tool. This Open Source agent framework from research at Nortel Network’s Harlow Laboratories implements the FIPA specifications about agent interoperability. FIPA-OS offers several advantages to the researchers: well-placed use of Java interfaces to separate agent subsystems, translation of incoming messages and system occurrences into events for internal processing, use of a conversation object to enforce protocols, intuitive use, easy inclusion of new agents and easy debugging. An important reason for the election of FIPA-OS is that of a thread for task (a task is defined as the primitive unit of work). This fact involves more robustness when comparing FIPA-OS with other tools that propose a thread for agent. As example, an agent can be available although a task could be blocked (for example in an infinite loop).

2.2 DAML+OIL language

DAML+OIL (DARPA Agent Markup Language + OIL), sometimes named simply DAML, is a semantic language markup designed by DARPA (Defense Advanced
2.3 Evenet2000

Research Projects Agency), that allows the definition of ontologies. It is based on W3C standardized languages such as RDF and RDFSP — so that XML developers do not find it excessively complicated to use-extend those languages with other tags, directly influenced by the description logic language OIL.

The authors have this language because it provides a basic infrastructure that allows a machine to make some sorts of simple inferences that human beings do (for example through the inverseOf, TransitiveProperty, samePropertyAs and complementOf properties) and has become a standard in ontology representations. As example, the following sequence of tags defines the class “Man” as a subclass of “Male”, disjoint with the “Woman” class, and has only a “Father” (technically a subclass of those classes that have cardinality 1 for the property “hasFather”).

In Section 3 more definitions of classes and axioms in that language are given. A more detailed description of DAML+OIL language can be found in [7]. At this moment a new, improved and more expressive markup language, known as OWL, is being developed. Nevertheless DAML+OIL, as can be seen in Section 3, is sufficiently expressive for carrying out the proposed task in MASCONTROL.

2.3 Evenet2000

The problem of optimization is much related to identification and control. In both disciplines, it is usually necessary to optimize some parameters for a model (identification) or for a controller. In the same way, the training of a neural network consists of finding the best values of the weights of the network. Because of this similarity of methods, the authors have considered the possibility of the application of neural networks training methods to control problems.

For this MAS, the authors have preferred used Evenet2000 [16], a Java-based neural network toolbox developed at the University of La Laguna — although most of the commercial NN trainers can be integrated in a MAS. This Open-source tool offers some features such as the possibility of designing an arbitrary neural network to optimize. This tool develops an approach to derive gradient algorithms for time-dependent neural networks, using the Signal Flow Graph theory and taking advantage of Object Oriented Programming. Moreover, the designed structure makes it not to be limited to algorithms based on the gradient. Currently, several learning algorithms (Gradient descent, random search, genetic algorithms…) and 1-D optimization methods for the learning rate (golden section, parabolic interpolation) have been included for three kinds of optimization problems (BP, RTRL and ANFIS). Moreover, Evenet-2000 allows users to include easily new neural network elements and algorithms in it. This tool has been shown as highly useful in a suite of control problems.

In MASCONTROL, Evenet2000 is used for two purposes. Firstly, for the optimization of the parameters of a model of the unknown system. Secondly, in the optimization of the parameters of a controller such as the value of \( K_p \), in a proportional controller. The different agents import Evenet2000 packages for its use.

III. ONTOLOGY DESIGN SUMMARY

The authors have decided to design a coarse-grained ontology [17], due to the relatively simpleness of its use in MASCONTROL. Defined classes are mainly related to control concepts: System, Input, Output, ReferenceInput, Error, ControlAction. A second group of classes are referred to the implemented optimization algorithms in Evenet2000: OptimizationMethod, TrainingMethod. As an example, the following DAML+OIL code defines NonLinearSystem as a subclass of System.

With respect to the defined properties, these are mainly referred to the values of defined classes: valueOfPole, valueOfZero, valueOfTransferFunction, valueOfControlAction. On the other hand, other properties are related to control concepts such as order and type of a system. These definitions allow the system to make some interesting inferences from some axioms defined in the ontology. For example, if a given system does not reach a desired reference input with a proportional control action, the MAS can deduce that the type of the system is 0. A suitable control action for a system like that is a proportional-integral one. Therefore, an agent can deduce that a proportional-integral control action is a suitable control action for a system that does not reach a desired reference input with a proportional control action (and when this axiom does not appear explicitly in the ontology). The following ontology extract defines these axioms. The tags are considered self-explained.
IV. MASCONTROL AGENT FRAMEWORK

The authors have treated this scenario through the agent framework shown in Fig. 2, composed of 8 different types of agents. Each agent assumes one of the roles that appear in a STR scheme.

ReaderCommandAgent (RCA)

This agent samples the output of the system. Another role assumed by this agent consists of calculating and sending the command to the system. Finally, it stores a vector containing input-output data. This vector makes possible the system identification and determines if the input is rich enough for that identification.

IdentificationAgent (IA)

Several identification agents can appear in the system. Each IA tries to identify the system from the input-output vector. For this purpose, it uses Evenet2000 modules. A system user or an IdentificationLoaderAgent (from a record of previous trainings) can select a training method for each IA. The system model is loaded as an Evenet2000 neural network file. In other words, for an IA, the problem is equivalent to the one of optimization of the weights of a neural network, whose training pattern is defined by the input-output vector. The authors have included a one-step ahead predictor in the identification process.

Several of these agents (referring even to different system models) can be launched in different computers, taking advantage of the possibility of parallelism provided by the MAS.

LinearIdentificationAgent (LIA)

Similar to the IA, but it assumes a model that allows a linear regression, for example an ARX model. In this sense, the model is defined by the orders of the numerator and the denominator of the transfer function. Instead of Evenet2000 modules, an object that implements identification through the Forgetting Factor technique is used.

CentralIdentificationAgent (CIA)

This agent manages the IAs (linear or not). Initially it asks the DF for the agents that offer the service of identification in the MAS. Every T seconds, the CIA asks the IAs for the results of the current optimizations, selecting that optimization with the best results. Then it informs the rest of the IAs with the same model the set of parameters that minimize the criterion function. This way, the IAs take this set as an initial training set for new optimization processes. With the aim of providing the MAS with some intelligence, the CIA counts how many times each training method is the best one optimizing the criterion function. This information is stored in the ontology. In future control processes, MASCONTROL could use this information for initiating other IAs.

OptimizerAgent (OpA)

This agent optimizes the controller parameters. For this purpose, it takes the set calculated by the identification agents and includes it as constants in a new system. This system is considered as a NN whose parameters are the controller ones. In a general way, patterns are chosen as pairs reference input, reference input in a series of different reference inputs, indicating that the system should follow the reference input as close as possible, penalizing high raising time and valued peaks. The model of the system can be easily changed due to Evenet2000 modularity. Like IAs, several OpAs can be launched in different computers, taking advantage of the possibility of parallelism.

CentralControlAgent (CCA)

Similar to CIA in the sense of a manager in the system, but this time related to the optimization of model parameters. Each T seconds, CCA asks CIA for the details (model with the best results, parameter set) of the identification. After analyzing these data, CCA asks OpAs for the parameters that minimize the criterion function and the value of this minimization. This agent stores the results for subsequent sending to the RCA. Finally, CCA orders OpAs to stop the current optimizations and to start a new one from Fig. 2. MASCONTROL agents framework.
the calculated optimal parameter set. As CIA, CCA stores the number of times that a given training method has been the best one for the analyzed control process.

**InputOutputAnalyzerAgent (IOAA)**

This is an optional agent that analyzes process input and output data (calculated by the RCA). This analysis is made in two ways. First, it tests, in an intuitive way, if the system input is rich enough. For this purpose, this agent calculates the maximum and minimum input value in the last N periods, and it tests if the subtraction of these values is less than a given low enough threshold. If that is the case, it is supposed that applied input is not rich enough and IOAA suggests RCA that the reference input should be changed. This change is supposed to improve the identification process. In a similar way, output data are analyzed too. In this case, IOAA could suggest a reference input change or a study about the type of the system. The option of changing the reference input can be inhibited on-line through a user interface.

**Ontology Agent (OA)**

This is one of the key considerations in this phase of the work. This way, it differs from other MAS-based control systems [3,4]. Currently, in MASCONTROL, this agent only takes part in the study of the type of the system.

It is important to remark that this framework can be used for every algorithm of identification and control. This description has been deliberately made as “black boxes”, giving mainly input-output information, as they are characterized by its behaviour and interactions. This is the basis of the Agent Oriented Programming (AOP) where the interactions are made by message interchange in contrast with the OOP, where these interactions are made by method invocation. Based on this feature, the designed systems are open, that is, different MAS from different developers can interact among them. The messages are expressed in FIPA-ACL, the FIPA standard for agent communication. As example, the following message is sent by the IOAA to the RCA requesting a change of reference input.

(request
 : sender (agent-identifier :name IOAA)
 : receiver (set (agent-identifier :name RCA)
 : content ( (action (agent-identifier :name RCA)
 (changeReferenceInput))
 : protocol fipa-request
 :language FIPA-SL)

**V. RESULTS**

Due to the transmission rate and optimization time (determined by the used network), MASCONTROL should be used for the controlling of not-excessively fast processes. In this context, the authors have checked the MAS control-ling of several and different plants. As an example, results referred to an interconnected tank system are described below. This is a SISO system, whose input is the water flow that goes into the input tank while the height in the second Tank \( h_2 \) is the system output (Fig. 3).

![Fig. 3. Interconnected tanks system scheme.](image)

The chosen plant is very simple, although the proposed method can be applied to more complex systems, with the limitations described above. Moreover, as can be seen, the control problem solved by the MAS is very easy. In this context, what is important in this paper is the fact of showing the potential that a controller, through the use of ontologies, can control systems using actions whose description, for example, is determined by rules and axioms defined by human beings, and can read on it (without knowing a priori) the logic of how to do the control.

Although the developed MAS does not necessitate the mathematical model of the system, it is important to see what kind of mathematical system the MAS control is capable of, in the specific example. In that case, the model of the system is a SISO system, whose input is the water flow that goes into Tank 1 \( q \), while height in Tank 2 \( h_2 \) is the system output.

Transfer function for this system can be modelled by the following equation:

\[
\frac{H_2(s)}{Q(s)} = \frac{R_i}{(1 + s \cdot R_i \cdot C_1)(1 + s \cdot R_2 \cdot C_2) + s \cdot R_2 \cdot C_1}
\]

where \( R_i \) is the “resistance” to the pass of water in the tank \( i \) and \( C_i \) is the section of the tank \( i \).

Experiments were carried using three different control actions: proportional, proportional-integral and pole replacement. For each control action, MASCONTROL used several OpAs and IAs optimizing different system models through different optimization methods. Regarding to proportional action controller, this control action is indicated to MAS in two points: in the implemented control action in RCA and in the model to optimize in the OpAs. In the case of proportional control action, this model consists of a closed-loop configuration with an only parameter to optimize: \( k_p \). As the process input is physically limited between 0 and 5 V, the model contains a non-derivability in the Criterion function. This fact implies that gradient-based training methods are not the most ap-
appropriate ones. With a initial value of $k_p = 1$, a learning phase is carried. In this phase reference input value is continually modified, looking for a better identification. When it is considered that identification is good enough, reference input is set to the desired value. Figure 4(a) shows an example of OpAs optimization with a variable reference (3, 1, 2, 0.8, and 1.5 V). In this sense, model behavior is supposed to be more independent from a given reference input. As it can be seen, the type of the system is zero since the system output does not reach the reference input. As it was expected from this fact, OpAs tend to increase $k_p$ since they try to minimize the cost. Figure 4(b) shows output evolution with a 3V reference input. In this figure, the two mentioned phases can be clearly distinguished. As can be seen the MAS is able to determine the best possible value for the parameter in the controller.

With respect to PI control action, the closed-loop model needs to be modified for including integral control action and the control action implemented by the RCA. These modifications are easy to carry due to MASCONTROL and Evenet2000 modularity. Figure 4(c) shows an example of OpAs optimization. In this case, the system output reaches the reference input in each section, presenting a low overshoot. This optimization is reflected in the system output: it reaches the desired reference input (3 V) and with a low overshoot (Fig. 4(d)). As can be seen the output provided by OpAs and system output present similar characteristics, so it can be considered as a validation test for the system.

A particular illustrative case about the use of an OA, and ontologies in general, in a control system is shown in Fig. 4(e). This figure shows the output evolution initially controlled by a proportional control action. When IOAA realizes that the system output is stabilized and that it has not reached the reference input, RCA asks OA for the suitable control action when a system does not reach a reference input using a proportional action. At this point, OA looks for this fact in the system ontology and informs RCA that the answer is a proportional-integral control action. Then, MASCONTROL makes the necessary modifications (control action and model). As it was expected, the process output reaches the desired reference input after these modifications. This fact is only an example of the power that ontologies are able to supply to a process control system.

Once the goodness of the agent architecture has been tested for system identification and control using proportional and proportional-integral control action, the following work focuses on studying that goodness for a totally different control action, in particular, pole replacement. This control action depends critically on system identification. Due to MAS modularity, some minor modifications are necessary. Firstly, RCA should implement this new control action. Secondly, controller parameter set optimization has no sense for this control action. Therefore, CCA is only limited to propagate CIA results. Figure 4(f) shows the results obtained. In this case, as it can be seen, the system identification (with PI control action) phase has been longer, looking for a better identification. In all these experiments, the system output reaches the desired reference inputs, although with some oscillations. Once more, the designed MAS has been shown as suitable for a more complex control action.

![Fig. 4. A. OpA optimization, P Control Action B. Proportional Control Action Results C. OpA optimization, PI Control Action, D. Proportional-Integral Control Action results, E. Proportional to proportional-integral control action change F. Pole Replacement Control Action Results.](image-url)
VI. CONCLUSIONS AND FUTURE WORK

A MAS architecture for process control is presented in this paper. In particular, this MAS implements a self-tuning regulator (STR) scheme [5], so it is important to remark that this paper has not offered a new general control algorithm but a new tool for its development. That is its main contribution: showing the potential that a controller, through the use of MAS ontologies, can control systems in an autonomous way, using actions whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the control. This is one of the main contributions of this paper.

For ontology design, DAML+OIL is a sufficiently expressive language for carrying out this work. Apart from the standard tools, the authors have included Evenet2000. This toolkit has been shown to be highly useful in control problems. The proposed MAS framework is composed of 8 different types of agents. Each agent assumes one of the roles that appear in a STR.

MASCONTROL experiments were carried using three different control actions: proportional, proportional-integral and pole replacement. For each control action, this MAS used several Optimization Agents and Identification Agents optimizing different system models through different optimization methods in several plants. As an example, results with respect to a tank system plant have been described, looking for the validation of the system. These results show the potential of the combination of MAS and ontologies in the control of process and automation. In this case, the main advantages of this approach are 1) the parallelism that MAS offers — allowing the distribution of calculations in different computers, 2) modularity that allows the inclusion on-line of different types of agents and 3) ontologies allow the interaction among agents designed by different developers and users can control systems in an autonomous way, using actions whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the control.

In contrast, this approach is not free of disadvantages compared to conventional techniques. The network and the implementation of the standard protocols imply that the approach can be used only in not excessively fast systems. The future work is addressed in the following points:

- Use of the Jess expert system in more complex algorithms
- Inclusion of mobile agents
- Migration of Evenet2000 files to a XML structure that it would allow a better interaction with the designed ontologies
- Migration of the ontology from DAML+OIL to OWL. This way, new tags can be used, for example owl:sameIndividualAs.
- Implementation of more complex control techniques, for example, Optimal Control.
- Application of the MAS to other fields such as Robotics and multi-robot systems.

The authors strongly recommend the use of MAS and ontologies in order to develop applications such as those analyzed in this paper. They expect to have shown that its use offers more than they cost.

REFERENCES