Development of a Code Generation System for Control Agents

JOSE AGUILAR, MARIELA CERRADA, FRANCISCO HIDROBO, FRANKLIN RIVAS, WILLIE ZAYAS
CEMISID, Facultad de Ingenieria
Universidad de los Andes
Nucleo La Hechicera, Merida
VENEZUELA

Abstract: - This work has as main goal, the development of a system that would allow the creation of control agents for the SCDIA, this includes the creation of an agent’s source code, its compilation and incorporation to the SCDIA. The SCDIA is a reference model for the development of Intelligent Distributed Control Systems based on Agents. The SCDIA has a Control Agents community consisting in 5 agents that resemble the elements of a closed Control Loop, these agents are: Coordinator Agent, Controller Agent, Measurement Agent, Acting Agent and Specialized Agent. Agent development Platform JADE was used for developing this System. The Platform follows the FIPA standards for multi-agent systems. The system has 3 main agents: Central Agent, Code Generator Agent and Behavior Agent. These Agents communicate with each other to generate the control agents of the SCDIA, through the use of a code generation ontology.

Key-Words: - Intelligent Distributed Control Systems, Multi-agent Systems, Code Generation System

1 Introduction

The agents arise as a response to the need of counting on software applications that would solve complex problems by minimizing the external intervention, by applying principles that would emulate human reasoning. The agents allow the creation of software systems with a higher adaptation capacity. The agent can be a pro-active and autonomous entity. These characteristics are crucial nowadays when it is necessary to manage and process huge quantities of unstructured information.

A system in which two or more agents interact is called a Multi-agents System (MAS). Taking the theory MAS as a principle, a reference model was created for the development of a Intelligent Distributed Control System Based on Agents (SCDIA), framed in the project Agenda Petróleo (Petrol Agenda), realized by the Universidad de Los Andes [1, 2, 3, 4]. The SCDIA counts on two main components: La Control Agents Community and the Service Management Agents Community. The agents that structure the Control Agents Community help each other to perform supervision and control tasks, related to Industrial Automation. Such community is composed by 5 agents which resemble the elements of a closed control loop, these agents are: Coordinator Agent, Controller Agent, Measurement Agent, Acting Agent and Specialized Agent. The Service Management Agents Community is formed by an Agents Administrator Agent, a Data Management Agent, an Applications Management Agent, a Resources Management Agent and a Communication Control Agent. In this work the creation System of Control Agents of the SCDIA is developed as a part of the project Agenda Petróleo, this includes the agent’s source code generation, its compilation and incorporation to SCDIA. For the development of this system, the JADE Agent Development Platform, which follows the FIPA MAS Standards, was employed. The developed system has 3 main agents: Main Agent, Code Generator Agent and Behavior Agent. Additionally a code generation ontology is proposed, which is necessary for these 3 agents. The Control Agents Generation System (SIGECO) of SCDIA simplifies the creation process of a new control agent, establishing the generic characteristics for agents, their behavior types, methods and their respective codes, apart from the use of external applications for specialized tasks. This allows the user to recycle the code and reduce the code writing for the creation of an agent. It should be highlighted that this component follows an open design to facilitate adding new functions.

The organization of this task is the following: the next section contains the theoretical aspects enclosing this task. Then the illustration of the SIGECO design using the MASINA methodology, which is a MAS specification methodology. The 4th section presents the implementation of SIGECO. Finally the conclusions and recommendations.

2 Theoretical Aspects

2.1 MAS and Agents

An agent is a software system placed in a certain environment and operates in a continuous cycle of
Perception-Reasoning-Actuation. The agent perceives the changes in its environment, applies the reasoning provided by already known and new available information and selects a plan of action in response. There are many classifications of agent types [11]:

**Reactive:** Acts in the event-condition-action mode. Answers only to external stimulus using the surrounding information available.

**Deliberative:** they hold knowledge about the domain in which they interact and the necessary planning capacity to accomplish a sequence of actions to finish up with a fixed task.

**Collaborative:** in general, in these cases the agents work together to solve a problem.

On the other hand, the MAS are characterized by the interaction of many agents in the same physical or virtual environment. One of the main concepts of the MAS is the interaction and coordination between agents, this is not limited only to the communication or message exchange, it is also includes the way in which an agent relates to other agents. [10].

### 2.2 MASINA

The specification of the agents using this methodology is based on the construction of the different models. The models are [5]:

- **Organization Model:** permits analyzing the organization where the MAS will be incorporated.
- **Agent Model:** describes the agents characteristics, their abilities, services, etc.
- **Task Model:** describes the tasks that will be carried out by the MAS, so that they can be distributed amongst the different agents that take part in it.
- **Intelligence Model:** shapes the capacity of generating an intelligent behavior on an agent.
- **Coordination Model:** describes all the coordination mechanisms between agents.
- **Design Model:** describes the architecture of the MAS, it is previous to the implementation.
- **Communication Model:** selects the information exchange between the different agents.

### 2.3 Intelligent Distributed Control System Based on Agents (SCDIA)

The SCDIA is a multi-agent platform designed for industrial automation systems [1,2,3,4]. It proposes a series of agents that represent the elements present in a control loop with the intention of establishing a generic mechanism for the organizations management related to industrial automation.

This model describes five types of agents associated to the coordination, measurement, control and other specialized tasks in an automation platform, these are grouped in a community labeled: Control Agents Community [1,2,4]. Furthermore, the model proposes another agents Community to provide the Control Agents Community with services and particularly, to administer the agents system and the computational platform where the system will come to life [3,4].

#### 2.3.1 The SCDIA’s Control Agents Community

The SCDIA proposes a control agents community which represents the components of a generic process control loop. The architecture of the platform proposes five types of Control Agents[1,2,4]:

- **Measuring Agent:** it is in charge of obtaining the necessary information to determine the state of the process.
- **Controller Agent:** it evaluates the information of the process and makes decisions that would allow to keep it in ideal state, productivity, quality and security conditions.
- **Acting Agent:** converts the decisions taken by the controller, coordinator or specialized agents in actions that bring up the necessary changes in the process for reaching the established tasks.
- **Coordinator Agent:** Supervises the control loop, plans the control schemes and decision making, produces changes in the controllers commands and even changes in the behavior of the control loop agents under its supervision.
- **Specialized Agent:** is an agent that carries out specific functions for system support for example, pattern recognition, failure diagnosis, etc.

#### 2.3.2 Service Management Agents Community (SMAC)

As a support for the Control agents community of the SCDIA, there is a group of management agents constituted by software components in a distributed environment. Each component can act as an access way for processing a determined application, as a bridge between remote clients and data sources, or as an access interface to resources and information systems. The SMAC is the heart of the distributed agents system, because it contains the agents that manage the communication services and confers characteristics to the system such as security, transparence, labeling, migration and interoperability. The SMAC is composed by five agents [3,4]:

- **Agents Administrator Agent:** it is in charge of managing, integrating and supervising the state of the SCDIA. This agent knows the location and state of all existing agents in the system.
- **Resources Management Agent:** distributes the elements necessary for the execution of any process,
as for example: processors, access/exit hardware, storage hardware, etc.

**Applications Management Agent:** this agent is in charge of locating the applications that could be required by an executing process, for example, numeric or symbolic calculus programs, artificial intelligence applications, etc.

**Data manager Agent:** this agent is in charge of establishing the link with sites where interest data for the executing process is found, coming from SCADA, databases, or any other hardware or application that could store data.

**Communication control Agent:** maintains and controls the communication between MAS. It is in charge of translate and manipulate ontologies.

### 3 SIGECO Design

In this work is proposed a creation system of control agents for SCDIA, called SIGECO. For this, the SCDIA gets a new community, called the Code creation Agents community (CGC), that compose SIGECO, which allow the generation of the source code of the control agents of SCDIA based on the control agents specification showed in [1,2].

SIGECO is made of 3 agents that collaborate with each other to carry out the creation, compilation and incorporation of the control agents for the SCDIA agents platform, which are: Main Agent, Code Generation Agent, and Behavior Agent. One code generation ontology is used by the CGC for communication.

#### 3.1 CGC Agents Description

**Central Agent (AC):** Collects information provided by the user concerning the agent to be created. These data include information about attributes, methods and behavior of the new agent. This information is used by the AC to generate the agent concept that will latter be transmitted to the AGC for creating the source code of the SCDIA. The code generation ontology defines the agent concept as the grouped attributes, links to libraries, methods, initiation codes and behaviors that also represent concepts. The source code of the creating agent comes from this information. This agent performs the creation request for the behavior source code to the AGS and the creation request of the agent source code to the AGC. The behaviors generated by the ACS are incorporated to the agent concept by the AC.  

**Code Generation Agent (AGC):** its task is to generate the source and object codes of the agent of the SCDIA. The AGC manages code patterns to generate the source code of an agent. Taking an agent concept as an input parameter, the AGC incorporates the information code that comes from the agent concept and generates the source code of an agent from the Control agents community of SCDIA. The communication between the CGC agents is based on the code generation ontology of the SCDIA. This agent generates the source code of an agent, obtains its object code and takes it into the agents platform. The source code is submitted to the AC to show it to the user.

**Behavior Agent (ACS):** generates the source code of the actions that an agent should take in a given moment. The ACS generates behavior source code from behavior patterns associated to the SCDIA control agents (each member of the agents community of the SCDIA has typical behaviors associated to their roles). Once the behavior source code is generated, it is showed to the user so that the necessary code lines are added up to adjust it to particular needs of the agent to be created.

#### 3.2 Tasks Model

The services and tasks of SIGECO are: a) Agent concept building Tasks: Adding Attribute, Adding Links to libraries, Adding Method, Adding parameters, Adding Initiation code, Adding Behavior, Create behavior source code, b) Code creation tasks: Creation of agent source code, Creation of agent object code, Incorporation of agents to SCDIA, c) Communication tasks: Request receiving, Transmission

#### 3.3 Coordination Model

The conversation that relates the agents of the CGC are shown in table 2.

<table>
<thead>
<tr>
<th>Initiation Agent</th>
<th>Involved Agents</th>
<th>Exchange</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>AC-AGC</td>
<td>Control Agent Creation request</td>
<td>Creation of Control agent source code (AGC).</td>
</tr>
<tr>
<td>AC</td>
<td>AC-ACS</td>
<td>Behavior code Creation request</td>
<td>Creation of Source code of the behavior model (ACS).</td>
</tr>
<tr>
<td>AC</td>
<td>AC-AGC</td>
<td>Verifies if the creation agent was successfully</td>
<td>Creation Source code of the Control agent (AGC).</td>
</tr>
</tbody>
</table>

Table 2. Exchange between the CGC agents.
We will explain a conversation. Check [6] to see the rest. The conversation “request for the creation of a control agent” is composed by the following: The AGC receives a message, in this case containing a request for the creation of a Control Agent, coming from the central agent. This request is answered once the agent is generated and incorporated to the platform of the SCDIA. This request contains the source code, created by the new agent.

The AGC receives the request and takes an action according to it. In this case, it creates the source and object codes and gets them into the SCDIA and returns the source code.

Figure 1. Request Conversation of the creation of a control Agent

3.4 Communication Model
Following the description of the conversation presented in the previous section we choose a speech act. Check [6] to see the rest. The speech act is: Create Agent Source Code. a) Objective: receive the agent creation request from the AC; b) Type: Directive; c) Communication: Direct; d) Participating agents: AC and AGC, e) Source: AC, f) Service: Control Agent Creation, g) Exchanged Data: Agent Concept, h) Description: The AGC receives the agent concept, necessary for the creation of the new control agent. i) Pre-condition: The AGC requires receiving the agent concept for generating the new control agent, j) Termination Condition: The agent concept is received by AGC, k) Performative: Request, l) Communication Media: computer network.

3.5 Code creation ontology of SCDIA
The ontology defines the concepts, actions and predicates used for the CGC to carry out the tasks related to the code creation. The source code of an agent oriented to the platform for Multi-agents system JADE, can be divided in code fragments with well defined functional characteristics, as it appears in figure 1. The conjunction of functional elements of an agent gives as result the agent concept. At the same time, each functional element has been defined as a concept, so that the agent concept is composed by the Attribute concepts, Links to Libraries, Initialization code, behavior and methods. Like that, the code creation ontology is composed by the following concepts, actions and predicates:

- **Concepts**: represent entities with complex structures which are part of the agent knowledge base, for example, an attribute is a complex entity that forms part of the source code of an agent. Others concept are: Links to libraries, Input parameter, Method, Behavior, Agent (represents the union of all the above concepts).

- **Predicates**: expressions that give an idea about the world’s state. The predicates can be employed to get to know the result of any action that has been requested by an agent to another.

- **Actions**: it is an activity that can be executed by an performing system, in this case, an agent. In our case, the actions are considered as equivalent to the services offered by each agent.

![Figure 2. Functional de composition of the source code of an agent.](image)

4 Development of SIGECO

4.1 Implementation of the CGC of SCDIA
SIGECO was developed using JADE [8]. The SIGECO agents are incorporated to the platform and interact through it with the purpose of generating the source code of the Control agents of the SCDIA. Afterwards, from this source code, the object code is obtained and the new agent is incorporated to the agents container of the SCDIA, this container could be local (in the same machine where the agent has been created) or remote (in another machine). The CGC is part of the SCDIA nucleus, for this reason Java packages were created in which the different elements of the CGC would be contained.
The kernel and ontology packages structure the nucleus of the system. The kernel package contains the different sections containing SCDIA agents, including the agents of the CGC. The ontology package assemble different sections that structure the ontology used by the CGC agents (and any other SCDIA agents) to achieve common objectives, in our case, for the creation of the control agent and its incorporation to SCDIA. Following, the sections description: a) Agent: contains the SCDIA agents, which means the agents of the Control Agents community of the SCDIA, the agents of the CGC and of the SMAC, b) General: contains the graphic presentation sections of the Central and Behavior agents c) Gui: Contains the graphic interface sections of the CGC through which information is shown or required to the user, d) Util: Contains utility method sections that carry out repetitive tasks which are executed throughout the whole application, e) Code-generation: assembles the sections and packages that form the code generation ontology, f) Action: contains the actions that can be requested to the CGC agents, g) Concept: contains the code generation ontology concepts, h) Predicate: Contains the predicates managed by the code generation ontology.

The main interface of SIGECO is given by the central agent (see figure 3).

Figure 3. Central Agent Interface of the CGC.

The behavior generator agent has also an interface (see figure 4), while the Code generator Agent does not have a graphic interface.

4.2 SIGECO main options

In the following we will describe some of the options of SIGECO, for more information see [6].

4.2.1 Agent Concept creation

The creation of a new agent requires the user to provide basic information as the agent name, attributes and methods. Afterwards the agent nature is defined through the incorporation of the agent behavior. The information provided by the user will allow the creation of the agent concept, with which the source code and the object of the new agent will be created. For the creation of the agent concept it is necessary amongst other things:

Add/Edit Links to Libraries: the links to libraries will allow access to external sections to the package in which the agent will be created.

Add/Edit agent methods: the methods provide the processing capacity to the agents and are able to carry out tasks such as, the call of methods from other sections that execute specific tasks which would serve the agents ends. The methods are part of the source code of every agent.

Add behavior to the agent: in JADE, an agent provides services through the behavior execution and defines its nature. The agents generator component counts with an interface to add behaviors to agents in the design stage. From this interface the user writes the source code of the behavior that the agent should have [6]. SIGECO counts with a behavior creator based on the Control agents types. The creator has pre-defined behaviors related to the tasks that the agents should execute according to their types.

Figure 4 presents the graphic interface of the behavior creator. For using it, selecting the type of agent for which the behavior is going to be designed and the type of behavior to be created.

4.2.2 Creation of a new Agent

The creation of the new agent is an easy process, it is enough to indicate the agent information and press the button “Create Agent” of the Central Agent interface, this will generate the new agent. Once the process is finished the interface will show the created agent’s source code (see figure 5).
5 Conclusion

This work has the specification of three new agents for the SCDIA, which are: Central Agent, Code Generator Agent and Behaviors Generator Agent. These agents are part of a new agents community of the SCDIA named CGC. The agents of the CGC permit the creation of the required agents to carry out control tasks in a process.

The code generation ontology can be enriched by adding continuous production related concepts and the relationships between its different components. Part of the future work with CGC agents creation is related to the incorporation of new predicates that would allow the checking of vital proprieties inherent to the SCDIA that is being created.

On the other hand, JADE is a platform for agents development based on the emphasis on behaviors associated to the agent that is being created. In our case, SIGECO has got predefined behaviors according to the control agent type, that should get extended to make a better use of the creating agents.

The B2MML standard proposed by the World Batch Forum (WBF) constitutes an important leap towards the criterion unification about the basic information which would be more representative of the production processes and which management by the CGC agents and the rest of the SCDIA agents will give to the creator agents a better capacity to integrate with other systems of a similar nature [12]. The code generation ontology of the SCDIA can be nourished from standards such as the B2MML.

Acknowledgements: this work was financed by the project 97003817 of the Petrol Agenda program of CONICIT, and by the project I-820-05-02-AA of CDCHT of the Universidad de Los Andes.

References: