Sistema multi-agente para crear agentes de control para el SCDIA

Multi-agent system to create control agents for the SCDIA

José Aguilar, PhD & Willie Zayas, Msc.
Centro de Estudios en Microelectrónica y Sistemas Distribuidos (CEMISID)
Universidad de Los Andes
aguilar@ula.ve

Recibido para revisión 30 de Marzo de 2008, aceptado 25 de Agosto de 2009, versión final 1 de Septiembre de 2009

Resumen— **Este trabajo tiene como objetivo desarrollar un sistema que permita la generación de los agentes de control del SCDIA, esto incluye la generación de código fuente del agente, su compilación y su incorporación al SCDIA. El SCDIA es un modelo de referencia para el desarrollo de Sistemas de Control Distribuido Inteligente basado en Agentes. El SCDIA propone una comunidad de Agentes de Control compuesta por 5 agentes, que se asemejan a los elementos de un lazo de control cerrado, estos agentes son: Agente Coordinador, Agente Controlador, Agente de Medición, Agente de Actuación y Agente Especializado. Para el desarrollo del sistema de generación de agentes de control, llamado SIGECO, se utilizo la plataforma de desarrollo de agentes JADE. SIGECO consta de 3 agentes: Agente Central, Agente Generador de Código y Agente de Comportamientos. Estos agentes se comunican entre sí para generar los agentes de control del SCDIA, mediante el uso de una ontología de generación de código.**

Palabras Clave—**Generación de Código, Sistemas Multi-agente, Agentes de Control, Sistemas de Control Distribuido.**

Abstract— **This work has like objective develop a system that allows the generation of control agents for the SCDIA, this includes the source code generation of the agent, its compilation and its incorporation to the SCDIA. The SCDIA is a framework for the development of Intelligent Distributed Control Systems based on Agents. The SCDIA proposes a community of Control agents composed by 5 agents, which represent the elements of a closed control loop, these agents are: Coordinating Agent, Controller Agent, Measurement Agent, Action Agent and Specialized Agent. For the development of the system of generation of control agents, called SIGECO, we have used JADE. SIGECO consists of 3 agents: Central agent, Code Generation Agent and Behaviors Agent. These agents communicate to each other to generate the control agents of the SCDIA, by means of the use of an ontology of code generation.**

Keywords— **Code Generation, Multi-agent System, Control Agent, Distributed Control Systems.**

I. INTRODUCCION

Los agentes surgen como respuesta a la necesidad de contar con aplicaciones de software que resuelvan problemas complejos minimizando la intervención externa, aplicando principios que emulen el razonamiento humano. Los agentes permiten crear sistemas de software con mayor capacidad de adaptación. El agente puede ser una entidad proactiva y autónoma. Estas características son cruciales hoy en día cuando es necesario manejar y procesar gran cantidad de información distribuida.

La teoría de agentes viene siendo muy usada en el área de control y automatización de procesos. Estos sistemas son complejos, distribuidos y persistentes, lo que hace muy natural el uso de la teoría de agentes en el diseño de estos sistemas. El uso de la teoría de agentes ha sido caracterizado por el establecimiento de los principios operacionales de automatización y control de proceso en dicha teoría. Algunos de los trabajos emblemáticos al respecto son los presentados en [12, 13, 17].

Tomando como principio la teoría de Sistema Multiagentes (SMA), se desarrolló un modelo de referencia para el desarrollo de Sistemas de Control Distribuido Inteligente basado en Agentes (SCDIA), en el marco del proyecto Agenda Petróleo llevado a cabo por la Universidad de Los Andes [1, 2, 3, 4, 8]. El SCDIA cuenta con dos componentes principales: La comunidad de Agentes de Control y la comunidad de Agentes de Gestión de Servicios. Los agentes que forman la comunidad de Agentes

de Control colaboran entre sí para llevar a cabo tareas de supervisión y control relacionadas con la automatización industrial. Dicha comunidad está compuesta por 5 agentes que se asemejan a los elementos de un lazo de control cerrado, estos agentes son: Agente Coordinador, Agente Controlador, Agente de Medición, Agente de Actuación y Agente Especializado. La comunidad de Agentes de Gestión de Servicios tiene la tarea de darle soporte a los agentes de Control para la realización de sus tareas. Dicha comunidad está conformada por un Agente Administrador de Agentes, un Agente Gestor de Datos, un Agente Gestor de Aplicaciones, un Agente Gestor de Recursos y un Agente de Control de Comunicaciones.

Como parte del proyecto Agenda Petróleo, en este trabajo se desarrolló un sistema que permita la generación de los agentes de control del SCDIA, llamado SIGECO, que incluye la generación de código fuente del agente, su compilación y su incorporación al SCDIA. Para el desarrollo de este sistema se utilizo la plataforma de desarrollo de agentes JADE, que sigue los estándares FIPA para SMA [9]. SIGECO consta de 3 agentes: Agente Central (MainAgent), Agente Generador de Código (CodeGeneratorAgent) y Agente de Comportamientos (BehaviourAgent). Adicionalmente, se propone una ontología para la generación de código para estos 3 agentes. Para el desarrollo de SIGECO se incorpora una nueva comunidad de agentes al SCDIA, compuesta por los 3 agentes antes nombrados. SIGECO simplifica el proceso de creación de un nuevo agente de control, al establecer características genéricas para los agentes, para tipos de comportamientos, para métodos, con sus respectivos códigos, además de la utilización de aplicaciones externas para tareas especializadas. Esto permite al usuario reutilizar código y minimizar la escritura de código para el agente a ser creado. Cabe destacar que este componente sigue una arquitectura abierta para facilitar la adición de nuevas funcionalidades.

Este trabajo se organiza como sigue: la siguiente sección contiene los aspectos teóricos que sirven de marco a este trabajo. Después se presenta el diseño de SIGECO usando la metodología MASINA, la cual es una metodología para especificar SMA [5, 6]. La sección 4 presenta la implementación de SIGECO. Finalmente se presenta un caso de estudio en el cual se muestran las funcionalidades de SIGECO. Finalmente se presentan las conclusiones y recomendaciones.

II. MARCO TEORICO

A. Agente, SMA y MASINA

Un agente es un sistema de software que se sitúa en un ambiente y que opera en un ciclo continuo de Percepción - Razonamiento - Actuación. El agente percibe los cambios en su entorno, aplica un razonamiento empleando la información conocida por él y la nueva información disponible, y selecciona una acción a tomar en consecuencia. Existen varias clasificaciones de tipos de agentes. Aquí presentamos una de ellas [18]: a) Reactivo: Actúa del modo evento-condición-acción. Responden sólo al estímulo externo usando la información de su entorno. b) Deliberativo: Tienen conocimiento del dominio en el que se desenvuelven y la capacidad de planificación necesaria para llevar a cabo una secuencia de acciones para lograr una meta fijada. c) Colaborativos: En general, en estos casos los agentes trabajan juntos para resolver un problema.

Por otro lado, los SMA se caracterizan por la interacción de varios agentes en un mismo entorno, ya sea este físico o virtual. Un concepto principal de los SMA es la interacción y coordinación entre agentes, que no se limita a la comunicación o al envío de mensajes entre estos, sino a la forma como un agente se relaciona con otros agentes. [16] propone como características de todo SMA las siguientes: a) Diseño: Los agentes que conforman un SMA pueden ser de distinta naturaleza desde el punto de vista del software o hardware que emplean en su funcionamiento; b) Entorno: El entorno en el cual se desenvuelve un agente puede ser estático (invariante en el tiempo) o dinámico (no estacionario); c) Percepción: En un SMA la información de entorno percibida por los agentes es distribuida, así los agentes pueden recoger datos que varían de acuerdo a la ubicación de éste, puede ser percibida en distintos momentos por los agentes o puede ser interpretada de distinta manera; d) Control: el control de un SMA es típicamente distribuido. No existe un sistema central que recolecte información del entorno y decida la acción que cada agente debe realizar; e) Comunicación: En un SMA la comunicación entre agentes es crucial. Generalmente esta comunicación está planteada en términos de envío y recepción de mensajes entre los agentes.

Por otro lado, MASINA es una metodología desarrollada en [5, 6] para el diseño de SMA para problemas específicos, la cual está basada en la metodología MAS-Common- KADS. La especificación de agentes usando esta metodología se basa en la construcción de los diferentes modelos. Las características de los modelos es la siguiente:

- Modelo de la Organización: permite analizar la organización donde el SMA será incorporado. Permite identificar los actores de la organización y sus casos de uso.
- Modelo de Agente: describe las características de los agentes, sus habilidades, servicios, etc.
- Modelo de Tarea: describe las tareas que serán realizadas por el SMA, de tal manera de distribuir las mismas entre los diferentes agentes que lo componen. Algunas de esas tareas podrán ser hechas usando técnicas inteligentes, tales como redes neuronales, algoritmos genéticos, etc.
- Modelo de Inteligencia: modela la capacidad de generar un comportamiento inteligente en un agente. Para ello, permite caracterizar la capacidad de almacenar conocimiento, razonar y aprender.
- Modelo de Coordinación: describe todos los mecanismos de coordinación entre los agentes, a través de los

cuales se establecen los procesos de trabajo colectivo, las negociaciones, etc. Así, este modelo permite caracterizar las conversaciones entre los agentes: los protocolos de comunicaciones directas e indirectas, las ontologías a usar, etc.

- Modelo de Diseño: describe la arquitectura del SMA, ocurre previo a la implementación.

- Modelo de Comunicación: detalla los intercambios de información entre diferentes agentes, es decir, los actos de habla entre agentes.

B. Sistema de Control Distribuido Inteligente basado en Agentes (SCDIA)

El SCDIA es una plataforma multiagentes diseñada específicamente para sistemas de automatización industrial [1, 2, 3, 4, 8]. Propone una colección de agentes que representan los elementos presentes en un lazo de control de procesos, con la intención de establecer un mecanismo genérico para el manejo de las actividades de organizaciones relacionadas con automatización industrial. Este modelo describe cinco tipos de agentes configurados para tareas de alto nivel, asociadas a la coordinación, control, medición y a tareas especializadas en la plataforma de automatización, agrupados en una comunidad que se denomina Comunidad de Agentes de Control. Además, el modelo propone otra Comunidad de agentes para proveerles servicios a los agentes de la Comunidad de Control, y en particular, para administrar el sistema de agentes y la plataforma computacional donde hará vida el sistema. La comunidad de agentes de control está compuesta por cinco tipos de agentes de control [1, 2, 4, 8]:

- Agente de Medición: Se encarga de obtener la información necesaria para determinar el estado del proceso. Actúa como recolector, consolidador y procesador de datos.

- Agente Controlador: Es un agente que evalúa la información del proceso y toma decisiones que permitan mantenerlo en el estado deseado y en las condiciones ideales de productividad, calidad y seguridad.

- Agente de Actuación: Convierte las decisiones tomadas por los agentes controladores, coordinadores o especializados, en acciones que producen los cambios necesarios dentro del proceso para alcanzar las consignas establecidas.

- Agente Coordinador: Supervisa el lazo de control, planifica los esquemas de control y toma de decisiones, produce cambios en las consignas de los controladores, e incluso cambios en el comportamiento de los agentes del lazo de control bajo su supervisión.

- Agente Especializado: Es un agente que lleva a cabo una función específica que apoya al sistema, por ejemplo reconocimiento de patrones, cálculos estadísticos, diagnóstico de fallas, etc.

La comunidad de agentes de gestión de servicios esta compuesta por cinco agentes, los cuales se describen a continuación [3, 4]:

- Agente Administrador de Agentes: se encarga de manejar, integrar y supervisar el estado del SCDIA. Este agente conoce la localización y estado de todos los agentes que existan en el sistema.

- Agente Gestor de Recursos: este agente se encarga de distribuir el uso de los dispositivos (hardware) necesarios en la ejecución de un proceso, como por ejemplo: procesadores, dispositivos entrada/salida, dispositivos de almacenamiento, etc.

- Agente Gestor de Aplicaciones: este agente se encarga de ubicar las aplicaciones que puedan ser requeridas por un proceso que se esté ejecutando, como por ejemplo, programas de cálculo numérico o simbólico, aplicaciones de inteligencia artificial, etc.

- Agente Gestor de Datos: este agente se encarga de establecer el enlace con los lugares donde existan datos de interés para el proceso que se esté ejecutando, sea que estos datos provengan de bases de datos (relacionales, orientadas a objetos, tiempo real, etc.), de SCADAS, DCS, medidores, o cualquier otro dispositivo o aplicación que pueda almacenar datos.

- Agente de Control de Comunicación: es el encargado de mantener y controlar la comunicación entre SMA. Se encarga de traducir y manipular ontologías, y mantener un estado confiable del canal de comunicación.

III. DISEÑO DE SIGECO

Como se señaló anteriormente, en este trabajo se propone un sistema de generación de agentes de control para el SCDIA, denominado SIGECO. Para desarrollar a SIGECO, al SCDIA se le incorpora una nueva comunidad, denominada "Comunidad de Agentes de Generación de Código" (CGC), compuesta por los 3 agentes de SIGECO. El diseño de los agentes de SIGECO se realizó siguiendo la metodología MASINA [5, 6]. Los 3 agentes de SIGECO son:

- MainAgent (Agente Central - AC).
- CodeGenerationAgent (Agente de Generación de Código - AGC).
- BehaviourAgent (Agente de Comportamientos - ACS).

Estos agentes permiten la generación de nuevos agentes a partir de los datos suministrados por el usuario. El procesamiento de estos datos da como resultado el código fuente de los agentes de Control. Este es compilado, y una vez

producido el código objeto, el nuevo agente es incorporado a la plataforma SCDIA. Una ontología de generación de código es utilizada por la CGC para comunicarse. Esta ontología permite a los agentes manejar conceptos relacionados con la composición del código fuente de un agente, su compilación y la incorporación de éste a la plataforma computacional del SCDIA.

A. Descripción de los agentes de la CGC

- Agente Central (AC): Su objetivo es recolectar información acerca del agente a generar. Funciona como recolector de información proporcionada por el usuario acerca del agente que se desea generar. Estos datos incluyen información acerca de atributos, métodos y comportamientos del nuevo agente, entre otros. Para esto, el AC posee una interfaz gráfica para capturar información del usuario. Esta información es empleada por el AC para generar el concepto de agente que luego será transmitido al AGC para generar el código fuente del agente del SCDIA. La ontología de generación de código define el concepto de agente como el conjunto de atributos, enlaces a librerías, métodos, código de iniciación y comportamientos, que a su vez representan conceptos. A partir de esta información se genera el código fuente del agente a generar. Este agente realiza la petición de generación de código fuente de comportamientos al ACS y la petición de generación del código fuente de un agente al AGC. Los comportamientos generados por el ACS son incorporados al concepto de agente por el AC. Los servicios que ofrece son: Agregar enlace a librerías de código, Agregar atributo de agente, Agregar método a un agente, Agregar comportamiento a un agente, Agregar código de inicio del agente, Solicitar la generación de código fuente de un agente.

- Agente de Generación de Código (AGC): Su objetivo es generar el código fuente y objeto del agente del SCDIA. El AGC maneja plantillas de código para generar el código fuente de un agente. Tomando como parámetro de entrada un concepto de agente, el AGC incorpora a las plantillas de código información proveniente del concepto de agente y genera el código fuente de un agente de la Comunidad de Agentes de Control del SCDIA. La comunicación entre los agentes del CGC está basada en la ontología de generación de código del SCDIA. Este agente genera el código fuente de un agente, obtiene su código objeto y lo incorpora a la plataforma de agentes. El código fuente es entregado al AC para que éste lo muestre al usuario. Los servicios que ofrece son: Generar código fuente del Agente, Generar código objeto del Agente e Incorporar nuevo agente al SCDIA.

- Agente de Comportamientos (ACS): Su objetivo es generar el código fuente de los comportamientos de los agentes de Control del SCDIA. Un comportamiento representa una acción que un agente debe realizar en un momento dado. El ACS genera código fuente de comportamientos a partir de patrones de comportamientos asociados a los agentes de control del SCDIA (cada miembro de la Comunidad de Agentes de Control del SCDIA tiene comportamientos típicos asociados a sus roles). Una vez generado el código fuente del comportamiento, éste es presentado al usuario para que agregue las líneas de código necesarias para ajustarlo a las necesidades particulares del agente que se desea generar. El servicio que ofrece es Generar código fuente de comportamientos.

B. Modelos de Tareas, Coordinación y Comunicación

Las tareas y servicios de los agentes de SIGECO son:

Tabla 1. Tareas de los agentes de SIGECO

• Tareas de construcción del concepto de agente
o Agregar atributo
o Agregar enlace a librerías
o Agregar método
o Agregar parámetro
o Agregar código de inicio
o Agregar comportamiento
o Generar código fuente de comportamiento
• Tareas de Generación de Código
o Generar código fuente de agente
o Generar código objeto de agente
o Incorporar agente al SCDIA
• Tareas de Comunicación
o Recibir solicitud
o Transmitir

Las conversaciones que involucran a los agentes de la CGC se presentan en la tabla 2.

Tabla 2. Conversaciones de los agentes de la CGC.

Agente Iniciador	Agentes Involucrados	Conversación	Servicio
AC	AC-AGC	Petición para generar Agente de Control.	Generar código fuente de Agente de Control (AGC).
AC	AC-ACS	Petición para generar código de comportamiento .	Generar código fuente de plantilla de comportamiento (ACS).
AC	AC-AGC	Pregunta acerca de si se generó agente con éxito.	Generar código fuente de Agente de Control (AGC).

Pasaremos a explicar una conversación, para el resto ver [7]. La conversación 'petición de generación de un agente de control' consiste en lo siguiente: El AGC recibe un mensaje, en este caso contentivo de una petición para generar un agente de control, proveniente del Agente Central. Esta petición es respondida una vez que es generado el agente e incorporado a la plataforma SCDIA. Esta respuesta contiene el código fuente generado para el nuevo agente.

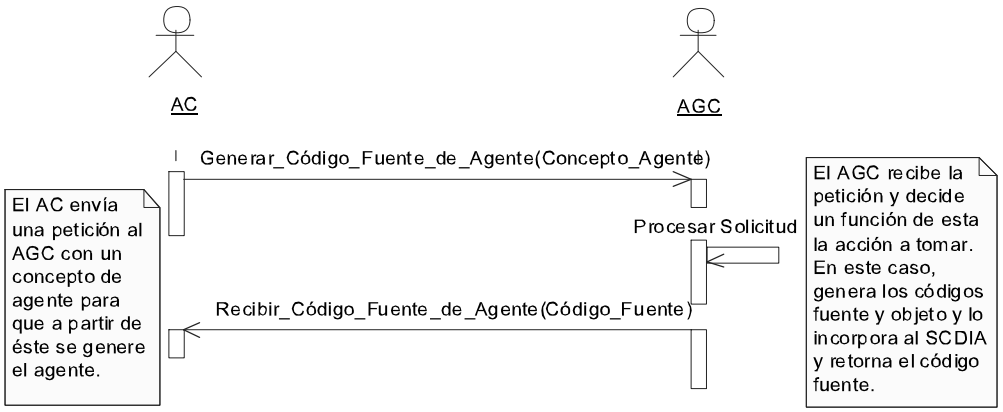

Figura 1. Conversación: "Petición de generación de un agente de control".

A continuación se describe uno de los actos de habla (modelo de comunicación) que compone la conversación presentada en la sección anterior, para el resto ver [7]. El acto de Habla es "Generar Código Fuente de Agente":

- Objetivo: Recibir solicitud de generación de agente por parte del AC.
- Tipo: Directivo.
- Comunicación: Directa.
- Agentes Participantes: AC y AGC.
- Emisor: AC.
- Servicio: Generar agente de control.
- Datos Intercambiados: Concepto de Agente.
- Descripción: El AGC recibe el concepto de agente necesario para generar el nuevo agente de control.
- Precondición: El AGC requiere recibir el concepto de agente para generar el nuevo agente de control.
- Condición de Terminación: El concepto de agente es recibido por AGC.
- Performativa: Petición (Request).
- Medio de Comunicación: Red de computadoras.

C. Ontología de Generación de código del SCDIA

Esta ontología define los conceptos, acciones y predicados utilizados por la CGC para llevar a cabo las tareas relacionadas con la generación de código. El código fuente de un agente orientado a la plataforma para sistemas Multiagentes JADE [11], puede dividirse en fragmentos de código con características funcionales bien definidas, tal como se observa en la figura 2. La conjunción de los elementos funcionales de un agente da como resultado el concepto de agente. A su vez, cada elemento funcional ha sido definido como un concepto, de manera tal que el concepto de agente está compuesto por los conceptos de Atributo, Enlace a librerías, Código de inicialización, Comportamientos y Métodos. Así, la ontología de generación de código está compuesta por los siguientes conceptos, acciones y predicados:

- Conceptos: Representan entidades con estructuras complejas que forman parte de la base de conocimiento de un agente, por ejemplo, un atributo es una entidad compleja que forma parte del código fuente de un agente. A continuación se detalla a cada uno de ellos:
- Concepto de Atributo: representa un atributo de la clase agente.
- Concepto de Enlace a Librerías: representa una sentencia que permite referenciar clases de paquetes que son externos al del agente.
- Concepto de parámetro de entrada: representa un parámetro de entrada para un método del agente y forma parte de la definición del concepto de Método.
- Concepto de Método: representa la declaración de un método que será invocado por el agente para cumplir las metas que el agente se haya puesto.
- Concepto de Comportamiento: representa uno de los conceptos más importantes, ya que el comportamiento del agente define su naturaleza. Los Comportamientos en la plataforma JADE definen la naturaleza de un agente. Todas las acciones que un agente debe ejecutar estarán contenidas en la codificación del comportamiento.
- Concepto de Agente: El concepto de agente representa la unión de todos los conceptos anteriores. Este concepto encapsula la información que el AGC requiere para generar el código fuente del agente. Así, se establece una

equivalencia entre el código fuente de un agente, en cada una de sus partes y los atributos contenidos por este concepto.

- Predicados: Expresiones que dan una idea acerca del estado del mundo. Los predicados pueden emplearse para conocer el resultado de alguna acción que ha sido solicitada por un agente a otro.
- Acciones: Es una actividad que puede ser ejecutada por un actor del sistema, en este caso, un agente. En nuestro caso, las acciones son consideradas equivalentes a los servicios que ofrece cada agente, los cuales fueron presentados en la sección 3.1.
- La ontología de generación de código puede ser enriquecida con funcionalidades adicionales a medida que las implementaciones del SCDIA maduren. Estas funcionalidades pueden incluir validaciones del concepto de agente al contrastarlo con modelos de procesos precargados en la CGC, o heurísticas que permitan comprobar si efectivamente el concepto satisface el fin que persigue el usuario, entre otras cosas.

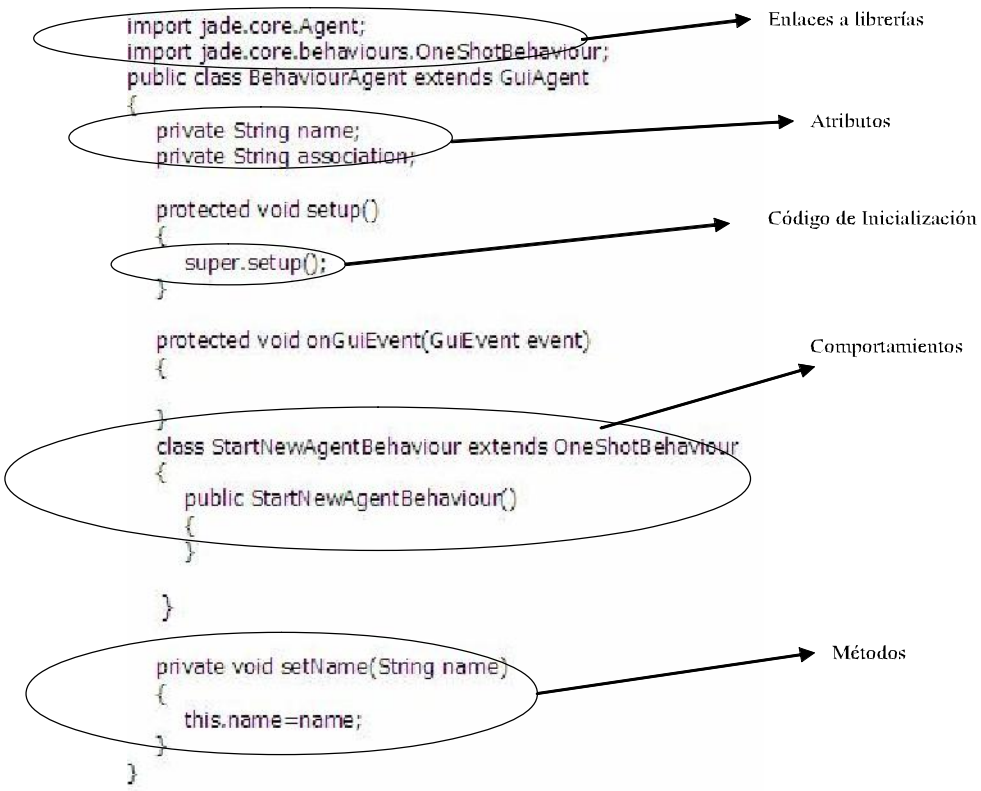

Figura 2. Descomposición funcional del código fuente de un Agente.

IV. DESARROLLO DE SIGECO

A. Implementación de la CGC del SCDIA

SIGECO fue desarrollado sobre la plataforma Multiagentes JADE [11]. Los agentes de SIGECO se incorporan a la plataforma JADE, y a través de ésta interactúan entre sí con el fin de generar el código fuente de los Agentes de Control del SCDIA. Posteriormente, a partir de este código fuente se obtiene el código objeto y el nuevo agente es incorporado al contenedor de Agentes del SCDIA, sea este local (en la misma máquina en la cuál ha sido generado el agente) o remoto (en una distinta). La CGC forma parte del núcleo del SCDIA, por esta razón se crearon paquetes de Java en los cuales los distintos elementos del CGC estarían contenidos. La figura 3 muestra la jerarquía de paquetes del SCDIA.

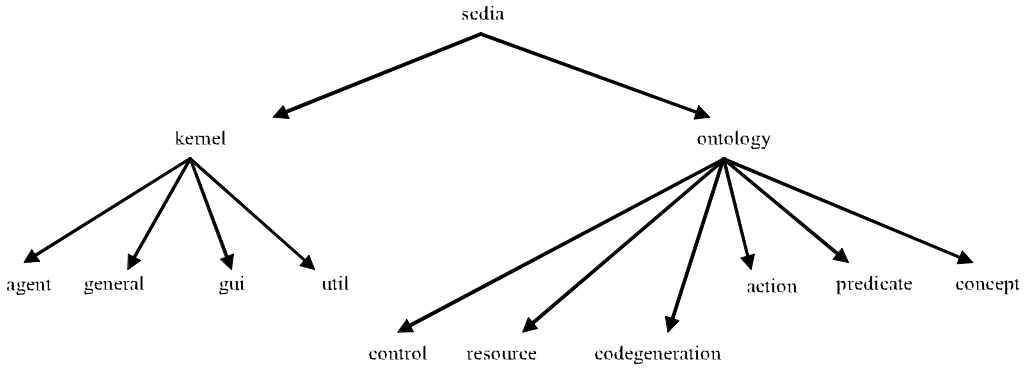

Figura 3. Jerarquía de paquetes del SCDIA.

Los paquetes kernel y ontology forman el núcleo del sistema. El paquete kernel contiene las clases que representan a los diferentes agentes del SCDIA, incluyendo ahora a los agentes de la CGC. El paquete ontology agrupa las clases que conforman la ontología empleada por los agentes del CGC (y cualesquiera agentes del SCDIA) para lograr objetivos comunes, en nuestro caso para la generación del agente de control y su incorporación al SCDIA. A continuación la descripción de las clases:

- Agent: Contiene los agentes del SCDIA, es decir a los agentes de la Comunidad de Agentes de Control del SCDIA, a los agentes de la CGC y a los agentes de la Comunidad de Agentes de Gestión de Servicios.
- General: Contiene las clases para la presentación gráfica de los agentes Central y de Comportamientos.
- Gui: Contiene las clases de interfaz gráfica de la CGC a través de las cuales se muestra o requiere información al usuario.
- Util: Contiene clases de utilidad con métodos que realizan tareas repetitivas que se ejecutan a lo largo de la aplicación.
- Codegeneration: agrupa las clases y paquetes que conforman la ontología de generación de código.
- Action: contiene las acciones cuya ejecución puede ser solicitada a los agentes del CGC.
- Concept: Contiene los conceptos de la ontología de generación de código.
- predicate: Contiene los predicados manejados por la ontología de generación de código.

La interfaz principal de SIGECO es dada por el Agente Central (ver figura 4).

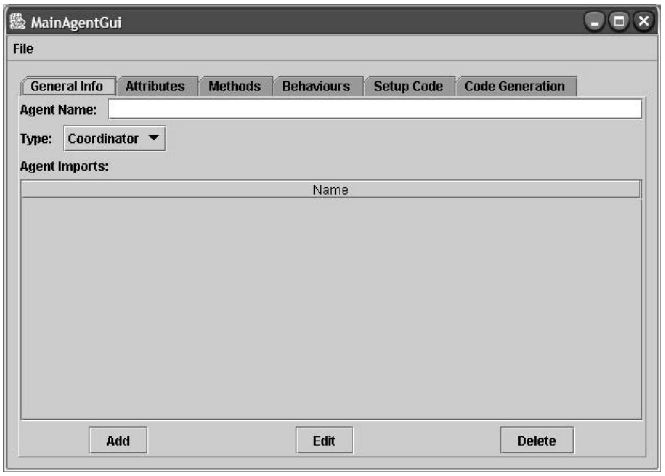

Figura 4. Interfaz del Agente Central del CGC.

El Agente Generador de Comportamientos tiene también una interfaz (ver figura 5), mientras que el Agente Generador de Código no posee interfaz gráfica.

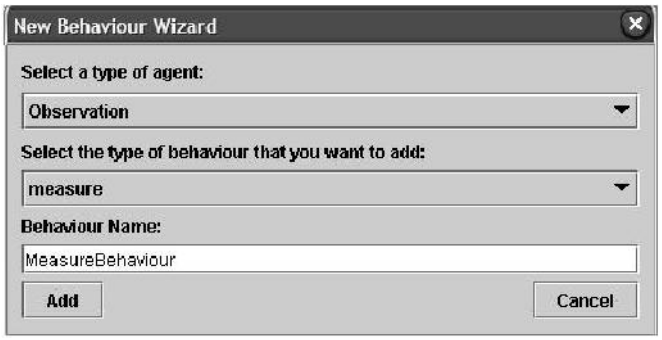

Figura 5. Interfaz gráfica del Agente Generador de Comportamientos.

B. Opciones principales de SIGECO

A continuación describiremos algunas de las opciones más importantes de SIGECO, para el resto ver [7].

B.1 Creación del concepto de agente

La creación de un nuevo agente requiere que el usuario proporcione información básica como nombre del agente,

atributos y métodos. Posteriormente se define la naturaleza del agente mediante la incorporación de comportamientos al agente. La información proporcionada por el usuario permitirá crear el concepto de agente, con el cual se generarán el código fuente y objeto del nuevo agente. Para la creación del concepto de agente se requiere, entre otras cosas:

- Agregar/Editar Enlaces a librerías: Los enlaces a librerías permiten acceder a clases externas al paquete en el cual será generado el agente. Para agregar un nuevo enlace a librerías haga clic en el botón "Add" bajo la lista de enlaces a librerías. La pantalla para agregar/editar enlaces a librerías aparece en la figura 6.

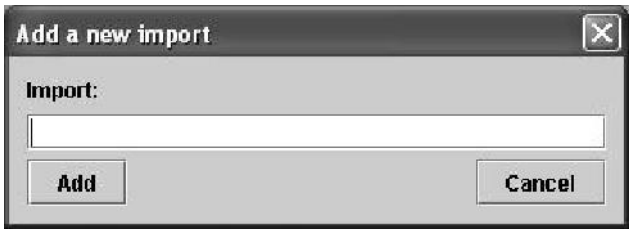

Figura 6. Cuadro de diálogo para agregar/editar enlaces a librerías.

- Agregar/Editar métodos del agente: Los métodos proporcionan capacidad de procesamiento a los agentes y pueden realizar tareas como por ejemplo, la invocación de métodos de otras clases que ejecuten tareas especificas que sirvan a los fines del agente. Los métodos forman parte del código fuente de todo agente. Los métodos pueden ser agregados a un agente a través de la lengüeta "Methods" de la interfaz gráfica del Agente Central. Para los métodos debe proporcionarse la siguiente información: Nombre del método, Tipo de retorno, Parámetros de entrada, y Cuerpo de código del método. La figura 7 muestra la pantalla para agregar/editar métodos.

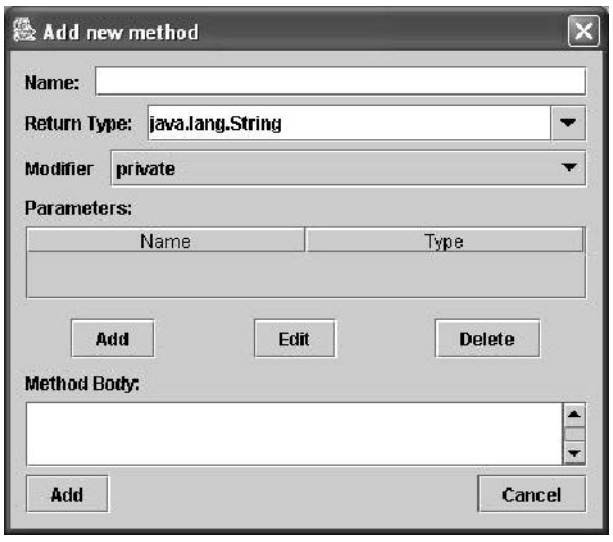

Figura 7. Cuadro de diálogo para agregar/editar métodos.

- Agregar/Editar comportamientos al agente: En JADE, un agente presta servicios mediante la ejecución de comportamientos, los cuales definen su naturaleza. El componente generador de agentes cuenta con una interfaz para agregar o editar comportamientos a los agentes en etapa de diseño (ver figura 8).

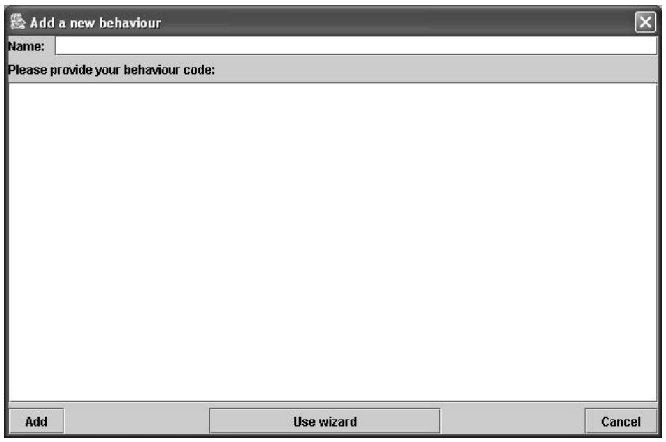

Figura 8. Cuadro de diálogo para agregar/editar comportamientos.

En el cuadro de diálogo de la figura 8, el usuario escribe el código fuente del comportamiento que desea que tenga el agente, si el mismo es nuevo, o edita un comportamiento existente. SIGECO cuenta con una librería de comportamientos basados en los tipos de agentes de control. Así, SIGECO tiene comportamientos predefinidos relacionados con las tareas que deben ejecutar los agentes según su tipo, para ser reutilizados por agentes bajo diseño [14]. La figura 5 presenta la interfaz gráfica del generador de comportamientos. Para emplearlo, basta con elegir el tipo de agente para el cual se va a generar el comportamiento, y el tipo de comportamiento a generar. Una vez hecho esto, el código es introducido en el cuadro de diálogo mostrado en la figura 8. Como se dijo antes, si el comportamiento ya existe, el mismo puede ser editado en el cuadro de diálogo mostrado en la figura 8

B.2 Generación de un nuevo agente

La generación del nuevo agente es un proceso sencillo, basta con haber indicado la información del agente y presionar el botón "Generate Agent" de la interfaz del Agente Central, esto generará al nuevo agente. Una vez realizado este proceso, la interfaz mostrará el código fuente generado para el agente (ver figura 9). El código fuente es almacenado en un archivo ".java", en el directorio raíz del generador de agentes.

Figura 9. Código fuente de un agente.

V. CASO DE ESTUDIO: SISTEMA DE CONTROL PARA UN PROCESO DE REFINACIÓN DE AZÚCAR

Las unidades de proceso que se muestran en la figura 10 forman parte del proceso de refinación de azúcar [15]. El proceso es alimentado de azúcar pura a través de una banda transportadora. Sobre el azúcar es rociada agua para formar sirope de azúcar. Este sirope es calentado en un tanque de dilución. De allí, el sirope fluye a un tanque de preparación en donde es calentado nuevamente y mezclado con otros componentes químicos. Desde el tanque de preparación el sirope fluye a un tanque de mezclado. A medida que el sirope fluye al tanque de mezclado se le agrega ácido fosfórico y lima. A los efectos de este caso de estudio, se considerarán como las variables a controlar:

1. Temperatura en el tanque de dilución.
2. Temperatura en el tanque de preparación.
3. Temperatura en el tanque de mezclado.

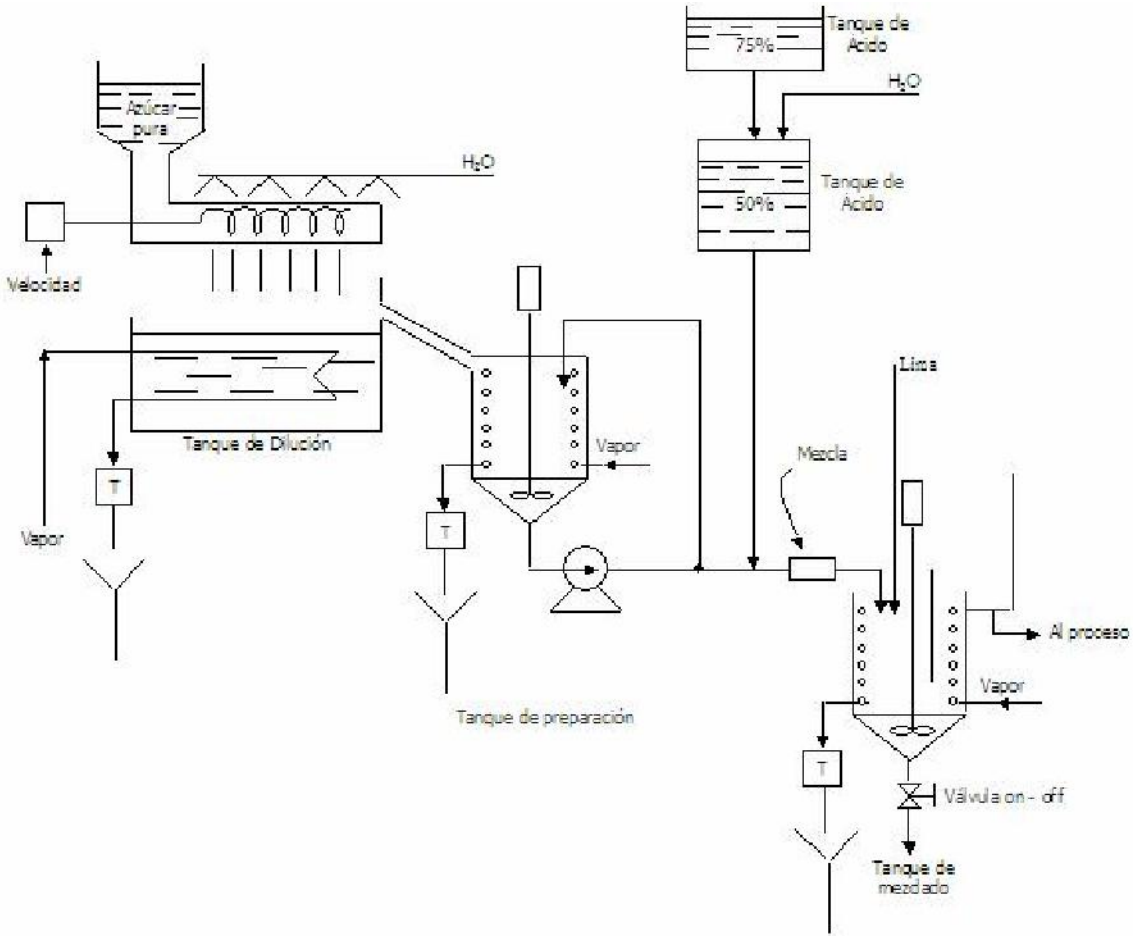

Figura 10. Proceso de refinación de azúcar [15].

A. Modelado del sistema

A.1 Medición de las temperaturas en los tanques de dilución, preparación y mezclado

Según la especificación del SCDIA, el Agente de Medición es el encargado de la adquisición de la información necesaria para conocer el estado del proceso. El caso de estudio requiere conocer las temperaturas en los tanques de dilución, preparación y mezclado, esta información es medida por sensores presentes en los tanques. Una vez realizado este proceso, la información es vaciada en archivos con formato XML que contienen el valor de la temperatura (en °C) y el instante de tiempo en el cual fue registrado ese valor. Los datos del proceso son analizados por el agente de medición, el cual emitirá una señal de alarma si el valor de la variable medida sobrepasa el límite establecido previamente. De ser este el caso, un mensaje de notificación será enviado al agente controlador, indicando el valor de temperatura que ha producido la señal de alarma.

A.2 Establecimiento de los parámetros de medición y control del proceso

El Agente Controlador debe asegurar que el proceso se mantenga en las condiciones requeridas. Para ello, el agente controlador debe conocer el estado del proceso y debe reaccionar ante cualquier cambio en las condiciones del sistema. El agente controlador recibirá información del proceso a través del agente de medición, éste le hará saber mediante una señal de alarma que las variables del sistema han salido del rango de comportamiento aceptable. El resultado de la recepción de esta señal es la obtención de una nueva acción de control, que en este caso de estudio, ajusta los parámetros de observación del Agente de Medición, estableciendo un nuevo rango de valores aceptables para la variable a medir.

El Agente Controlador deberá proporcionar información de desempeño del proceso al Agente Coordinador, por expresa petición de éste. Esta información incluye los datos de las señales de alarma recibidas, tales como: emisor de la señal de alarma, valor de temperatura e instante de tiempo en el cuál se recibió la señal, así como la totalización de las señales de alarma recibidas.

B. Desarrollo de los agentes utilizando el CGC

B.1 Agentes de Medición

El CGC puede generar comportamientos típicos del agente de medición tales como medición de variables, cambio en los parámetros de medición y envío de señales de alarma. Para lograr la comunicación con agentes del nivel superior, tales como el Agente Controlador, el Agente de Medición incorpora un comportamiento de búsqueda de Agentes Controladores. Para cada uno de los Agentes de Medición se crearon 4 comportamientos:

- MeasureBehaviour: Este comportamiento representa el proceso de medición de las variables del proceso. La medición de las temperaturas de los tanques de dilución, preparación y mezclado se realiza cada cierto tiempo y son contrastadas con el rango de temperatura aceptable. Este comportamiento es cíclico, por lo tanto se ejecuta de forma continua durante el tiempo en el cual el agente se encuentra activo, realizando mediciones en intervalos de tiempo de 5 segundos para el tanque de dilución, 1 segundo para el tanque preparación, y 2.5 segundos para el tanque de mezclado.

- SetMinMaxValueBehaviour: Este comportamiento permite establecer el rango de valores permitidos para una variable, en nuestro caso de estudio la variable temperatura.

- ReceiveMessagesBehaviour: Este es un comportamiento cíclico que recibe mensajes provenientes de otros agentes. Este comportamiento recibe las ordenes de cambio en los parámetros de medición.

- SendAlarmBehaviour: Este comportamiento es del tipo atómico (se ejecuta una sola vez). El Agente de Medición lo ejecuta cuando la variable medida ha salido del rango de valores permitido. Envía una señal de alarma que contiene el nombre del agente emisor de la señal, el valor de la temperatura y el instante de tiempo en el cual se ha producido la medición de ésta.

B.2 Agente Controlador

El Agente Controlador en este caso de estudio recibe señales de alarma provenientes de los Agentes de Medición, establece el nuevo rango aceptable para la temperatura medida en los tanques, y procesa solicitudes de información por parte del Agente Coordinador. Los comportamientos generados para el Agente Controlador son los siguientes:

- ReceiveMessagesBehaviour: Este es un comportamiento cíclico que permite recibir mensajes enviados por otros agentes del sistema de control. En el caso particular de este agente, las solicitudes de información provenientes del Agente Coordinador y las señales de alarma emitidas por los Agentes de Medición.

- SetMeasureParametersBehaviour: Este comportamiento envía al Agente de Medición una orden de cambio en el rango de los parámetros de medición.

- SendProcessInformationBehaviour: Este proceso envía al Agente Coordinador un informe del estado del proceso. Este informe contiene el total de señales de alarma recibidas, así como detalles de estas señales, tales como: Agente Emisor e instante de tiempo en el cual se recibió la señal de alarma.

B.3 Agente Coordinador

Siendo el Agente Coordinador un agente de jerarquía superior en el proceso de control, éste solicita información acerca del estado del proceso. Este caso de estudio se concentró en la petición de información del proceso al Agente Controlador. Cada

15 segundos, el Agente Coordinador envía una solicitud de información al Agente Controlador que éste debe responder. Los comportamientos incorporados al Agente Coordinador son:

- ReceiveMessagesBehaviour: Este comportamiento cíclico recibe los mensajes enviados por el o los agentes controladores del sistema de control.

- RequestInformationBehaviour: Este comportamiento de tipo periódico solicita cada 15 segundos información acerca del proceso al Agente Controlador. Esta solicitud es procesada por este agente y una respuesta es enviada en consecuencia.

C. Análisis de Resultados

Se pudo comprobar que efectivamente, los agentes generados mediante el CGC cumplieron con las tareas establecidas en los comportamientos definidos para cada agente, a través de una revisión exhaustiva del código generado. Dichos agentes constituyen la Comunidad de Agentes de Control del SCDIA. La figura 11 se obtuvo gracias al agente utilitario Sniffer de JADE, que permite observar el envío de mensajes entre agentes que se encuentren activos en JADE en un momento dado. Tal como lo muestra la figura, puede comprobarse el pase de mensajes entre los agentes de control del caso de estudio.

Una de las características más poderosas de la plataforma JADE es la incorporación dinámica de comportamientos. Gracias a esta característica es posible incorporar comportamientos al agente de acuerdo a las condiciones del entorno u otras consideraciones. Es esta característica la que permite la modificación, en tiempo de corrida, del rango de valores aceptables para cada Agente de Medición.

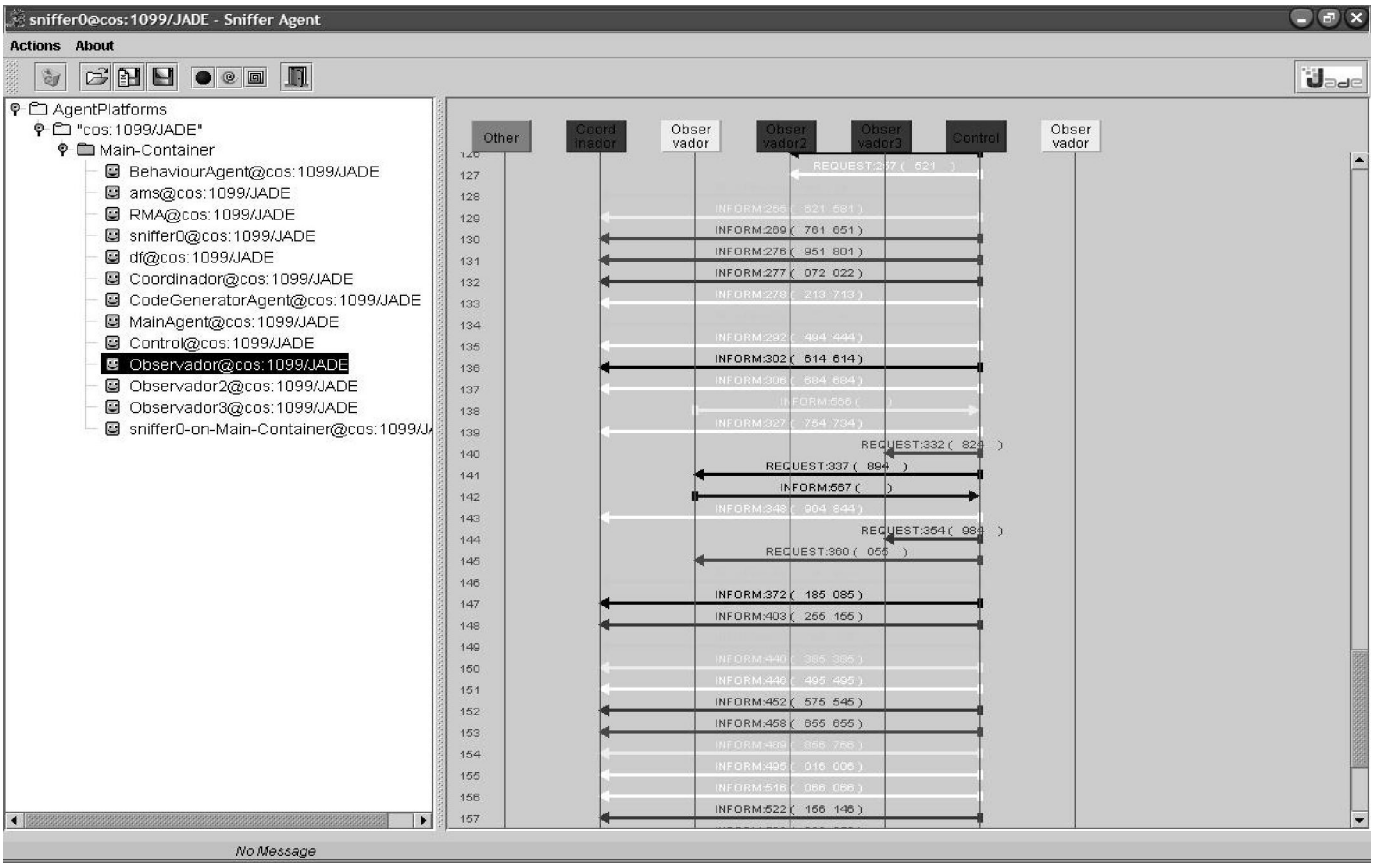

Figura 11. Pase de mensajes entre los agentes del caso de estudio.

VI. CONCLUSIONES

El desarrollo de este trabajo dio como resultado la especificación de tres nuevos agentes para el SCDIA, a saber: El Agente Central, el Agente Generador de Código y el Agente de Comportamientos. Estos agentes forman parte de una nueva comunidad de agentes para el SCDIA denominada Comunidad de Generación de Código (CGC) del SCDIA. Estos agentes colaboran entre sí para generar a los agentes de la comunidad de agentes de control del SCDIA. Así, los agentes del CGC permiten generar los agentes requeridos para realizar tareas de control de procesos. Ahora bien, el código fuente de los agentes generados debe ser personalizado, pero el tiempo para su puesta en ejecución se ve reducido porque el CGC contiene los comportamientos básicos clásicos para cada agente.

La ontología de generación de código resultante de esta investigación, maneja los conceptos básicos para la generación de agentes de control, así como acciones y relaciones

que involucran estos conceptos. Esta ontología puede ser enriquecida agregando conceptos relacionados con procesos de producción continua y las relaciones entre sus distintos componentes. Parte del trabajo futuro de desarrollo de los agentes del CGC está relacionado con la incorporación de nuevos predicados que permitan verificar propiedades vitales inherentes al SCDIA que se está creando.

Por otro lado, JADE es una plataforma para el desarrollo de agentes basada en el enfoque de comportamientos asociados al agente que se este diseñando. Esto permite definir la naturaleza de un agente en tiempo de diseño, y su modificación agregando o eliminando comportamientos del agente en tiempo de corrida. En nuestro caso, SIGECO dispone de clases de comportamiento predefinidos según el tipo de agente de control, que deberán extenderse para dar mayor utilidad a los agentes que sean generados.

El estándar B2MML propuesto por el World Batch Forum (WBF) [10, 19] constituye un paso importante en la unificación de criterios acerca de la información básica más representativa de procesos de producción, y cuyo manejo por parte de los agentes del CGC y el resto de los agentes del SCDIA dará a los agentes generados mayor capacidad para integrarse con otros sistemas de naturaleza similar, así como para compartir información con estos. La ontología de generación de código del SCDIA puede enriquecerse a partir de estándares como el B2MML para incorporar información estandar sobre los procesos a los agentes del CGC y a los agentes generados por estos. De esta manera, se pueden crear agentes de control más integrados y conscientes del proceso industrial en el cual interactuarán en tiempo de corrida. Conceptos tales como Segmentos de Proceso y Materia prima, definidos en este estándar, pueden ser de gran utilidad para los agentes de control.

Reconocimiento -Este trabajo fue financiado por el Proyecto 97003817: "Esquemas generales basados en Sistemas Inteligentes para Control de Procesos", del programa Agenda Petróleo del CONICIT, y por el CDCHT-ULA (proyecto I-820-05-02-AA).

REFERENCIAS

[1] Aguilar J., Cerrada M., Hidrobo F., Mousalli G. y Rivas F., 2005. A Multiagent Model for Intelligent Distributed Control Systems, En: Lecture Notes in Artificial Intelligence, Vol. 3681, pp. 191-197.

[2] Aguilar J., Cerrada M., Diaz H., Hidrobo F., Mousalli G., Rivas F. 2001, Application of the Agent Reference Model for Intelligent Distributed Control Systems. En: Advances in Systems Science: Measurement, Circuits and Control (Ed. N. Mastorakis, L. Pecorelli), WSES Press, pp. 204-210.

[3] Aguilar J., Cerrada M., Bravo V., Díaz H., 2004. Diseño de un Medio de Gestión de Servicios para Sistemas Multiagentes, En: Actas de la XXX Conferencia Latinoamericana de Informática, pp. 431-439.

[4] Aguilar J., Bravo C., Rivas F. 2004. Diseño de una Arquitectura de Automatización Industrial basada en Sistemas Multiagentes, En: Revista Ciencia e Ingeniería, Facultad de Ingeniería de la Universidad de los Andes, Vol. 25, No. 2, pp. 75-88.

[5] Aguilar J., Vizcarrondo J., Perozo N. 2006. Definition of a Verification Method for the MASINA Methodology, En: International Journal of Information Technology, Vol. 12, N.3, pp. 121-131.

[6] Aguilar J., Cerrada M., Hidrobo, F. 2007. A Methodology to Specify Multiagent Systems. En: Lecture Notes in Artificial Intelligence, Vol. 4496, pp. 92-101.

[7] Aguilar J., Zayas W., 2007. Desarrollo de un Componente de Software para los Agentes de Control del SCDIA, Informe Técnico del CEMISID # 21-2007, Facultad de Ingeniería, Universidad de los Andes, Mérida, Venezuela.

[8] Cerrada M., Cardillo J., Aguilar J., Faneite R., 2007. Agents-Based design for fault management systems in industrial processes, En: Computer in Industry, Vol. 58, pp. 313-328.

[9] FIPA Abstract Architecture. http://www.fipa.org/specs/fipa00001/SC00001L.html#_Toc26668589 (consultado marzo de 2009).

[10] Instruments Society of America: http://www.isa.org.

[11] JADE: http://www.telecomitalialab.com (consultado marzo de 2009).

[12] Jennings N., Bussmann S., 2003. Agent-based Control Systems. En: IEEE Control Systems Magazine, pp 87-94.

[13] Pinto J., 2000. Instrumentation & control on the frontiers of a new millennium. En: Journal Instruments & Control Systems, Vol. 1, pp. 78-90.

[14] Sametinger J., 1997. Software Engineering with Reusable Components. Berlin: Springer-Verlag.

[15] Smith C., 1985. Principles and Practice of Automatic Process Control. New York: John Wiley & Sons.

[16] Vlassis N., A concise introduction to Multiagent Systems and Distributed AI. University of Amsterdam. 2003: http://carol.science.uva.nl/~vlassis/cimasdai/cimasdai.pdf. (consultado marzo de 2009)

[17] Wagner T., 2003. Applying Agents for Engineering of Industrial Automation Systems. En: Lecture Notes in Artificial Intelligence. Vol. 2831, pp. 62-73.

[18] Weiss G., 1999. MultiAgent Systems. Cambridge: The MIT Press.

[19] World Batch Forum: http://www.wbf.org. (consultado marzo de 2009).
http://www.wbf.org/XML_Markup_Languages/B2MML/b2mml_intro.htm.

Professor Aguilar Jose received the B. S. degree in System Engineering in 1987 (Universidad de Los Andes- Venezuela), the M. Sc. degree in Computer Sciences in 1991 (Universite Paul Sabatier-France), and the Ph. D degree in Computer Sciences in 1995 (Universite Rene Descartes-France). He was a Postdoctoral Research Fellow in the Department of Computer Sciences at the University of Houston (1999-2000). He is a Titular Professor in the Department of Computer Science at the Universidad de Los Andes. He has published more than 200 papers and 5 books, in the field of parallel and distributed systems, computational intelligence, science and technology management, etc. Dr. Aguilar has been a visiting research/professor in different universities and laboratories, has been the coordinator or inviting research in more than 20 research or industrial projects, and has supervised more than 20 M. S. and Doctoral students in their thesis.