UNIVERSIDAD CATÓLICA DE SANTA MARIA

ESCUELA DE POSTGRADO

MAESTRÍA EN INGENIERÍA DE SISTEMAS

MODELO DE COMUNICACIÓN PARA SISTEMAS MULTIAGENTE UTILIZANDO WEB SERVICES

Tesis presentada por el Bachiller:
GUILLERMO ENRIQUE CALDERÓN RUIZ

Para optar el Grado Académico de:
MAGÍSTER EN INGENIERÍA DE SISTEMAS

AREQUIPA – PERÚ
2009

Presentación

Señor Director de la Escuela de Postgrado

Señores Miembros del Jurado

De conformidad con las disposiciones del reglamento de Grados y Títulos de la Escuela de Postgrado, pongo a vuestra consideración el presente trabajo de investigación titulado: "MODELO DE COMUNICACIÓN PARA SISTEMAS MULTIAGENTE UTILIZANDO WEB SERVICES". El mismo que de ser aprobado me permitirá obtener el grado de Magíster en Ingeniería de Sistemas.

Dedicatorias

Para ti Sonita,

que me miras y ayudas desde el cielo

Para Nancy y María Gracia,

los pilares que me hacen seguir adelante

Epígrafe

Intentar analizar problemas con información incompleta y comunicación deficiente generará más problemas que soluciones (Gerhard Weiss)

Índice general

Índice general
Índice de figuras 06
Índice de tablas 07
Resumen 08
Abstract 08
Introducción 10

CAPÍTULO ÚNICO: RESULTADOS 11
1. MODELO DE COMUNICACIÓN UTILIZANDO WEB SERVICES 11
1.1. Modelo Propuesto 12
1.2. Diseño del Modelo 14
1.2.1. Protocolo de interacción 14
1.2.2. Lenguaje de comunicación 19
1.2.3. Protocolo de transporte 22

2. IMPLEMENTACIÓN DEL SIMULADOR 24
2.1. Análisis 25
2.2. Diseño 265
2.2.1. Diagrama de clases 26
2.2.2. Servicios web 26
2.2.3. Pizarra 27
2.2.4. Ambiente de simulación 28
2.3. Implementación 30
2.4. Funcionamiento 30

Conclusiones 31
Sugerencias 33
Propuesta de desarrollo 34
Bibliografía 38

Anexo 1: Proyecto de Tesis 39
Anexo 2: Evaluación del modelo de comunicación 81
Anexo 3: Guía de funcionamiento del simulador 88
Anexo 4: Código de implementación del simulador 92
Anexo 5: Tabulación de resultados de las encuesta 97

Índice de Figuras

Figura 1: Componentes básicos de la comunicación 12
Figura 2: Modelo de comunicación propuesto 13
Figura 3: Interacción asistida de los agentes 14
Figura 4: Elementos constituyentes de Protocolo de interacción 15
Figura 5: Protocolo de interacción del Modelo de Comunicación propuesto 17
Figura 6: Diagrama de clases de la pizarra 17
Figura 7: Interrelaciones del protocolo de interacción 19
Figura 8: Solicitud y respuesta en SOAP 20
Figura 9: Estructura de un mensaje SOAP 21
Figura 10: Solicitud y respuesta en HTTP POST 21
Figura 11: Respuesta del protocolo HTTP POST 22
Figura 12: Diagrama de interacción del simulador 25
Figura 13: Diagrama de clases del simulador 26
Figura 14: Modelo Entidad-Relación de la Base de Datos de la Pizarra 28
Figura 15: GUI principal del simulador 29
Figura 16: GUI que muestra una solución al proceso de construcción de una casa 29
Figura 17: Agente inteligente 43
Figura 18: Áreas de la Inteligencia Artificial Distribuida 45
Figura 19: Arquitectura genérica de un MAS 47
Figura 20: MAS con agentes homogéneos no comunicativos 49
Figura 21: MAS con agentes heterogéneos no comunicativos 49
Figura 22: MAS con agentes homogéneos comunicativos 50
Figura 23: MAS con agentes heterogéneos comunicativos 51
Figura 24: Interacción baja 55
Figura 25: Interacción media 55
Figura 26: Interacción alta 56
Figura 27: Componentes básicos de la comunicación 60
Figura 28: Comunicación con coordinación asistida 62
Figura 29: MOCOSMA pregunta 1. ¿Cómo considera Ud. el nivel de entendimiento del modelo de comunicación? 81
Figura 30: MOCOSMA pregunta 2. Después de utilizar el simulador, ¿le quedó claro el modelo? 82
Figura 31: MOCOSMA pregunta 3. Desde su perspectiva, ¿es sencillo el modelo planteado? . 83
Figura 32: MOCOSMA pregunta 4. Basándose en su experiencia, ¿cómo considera Ud. la implementación del modelo de comunicación? 84
Figura 33: MOCOSMA pregunta 5. ¿Considera Ud. que el modelo tiene todo lo necesario para complementar la implementación de un sistema multiagente? 85
Figura 34: MOCOSMA pregunta 6. ¿Cree Ud. que el modelo se adapte y sea entendido por todo tipo de usuario? 86
Figura 35: MOCOSMA pregunta 7. Si su respuesta anterior fue sí, responda la pregunta ¿cómo cree Ud. que será esta adaptación? 87
Figura 36: Pantalla principal del simulador. 88
Figura 37: Información del Proceso de elaboración de un libro, que muestra las actividades asociadas y sus respectivas restricciones. 89
Figura 38: Ejecución del proceso Construir Casa. 90
Figura 39: Mensaje 90
Figura 40: Seguimiento de la ejecución del proceso 90
Figura 41: Solución del proceso de elaboración de un libro 91

Índice de Tablas

Tabla 1: Descripción de las clases del Protocolo de Interacción 15
Tabla 2: Descripción de las componentes de la pizarra 18
Tabla 3: Descripción de atributos de los componentes de la pizarra 18
Tabla 4: Características generales del sistema multiagente simulado 24
Tabla 5: Tareas y restricciones de los procesos del sistema multiagente simulado 24
Tabla 6: Descripción de atributos de las clases 27
Tabla 7: Características técnicas del Simulador de Modelo de Comunicación 30

Resumen

La implementación de Sistemas Multi-Agente (MAS, MultiAgent Systems) es muy difícil y escaza en nuestro medio, debido a la complejidad de las metodologías y modelos existentes. Uno de los modelos más importantes de un MAS es el de comunicación. Este modelo presenta una complejidad aun mayor, debido a la necesidad de implementar un lenguaje común y protocolos de transporte e interacción. Es por esta razón que se hace necesario contar con un modelo de comunicación que facilite la implementación de un MAS.

En el presente trabajo se propone un modelo de comunicación para Sistemas Multiagente utilizando web services, el cual pretende suplir la carencia mencionada. Se presenta también un simulador, el cual ha sido desarrollado para clarificar el funcionamiento del modelo.

Luego de presentar el modelo a un grupo de desarrolladores, se ha podido demostrar que el modelo es entendible, implementable y adaptable a todo tipo de usuario, con una aceptación muy cercana al 100%.

Abstract

Implementation of Multi-Agent Systems (MAS) is very difficult and scarce today, due to the complexity of existing models and methodologies. One of the most important models is the MAS communication model. This model presents even more complexity because of the need to implement a common language, transport and interacting protocols. That is why it is necessary to have a communication model that facilitates the implementation of MAS.

In this Thesis, a communication model for Multiagent Systems using Web services is proposed, which aims to fill the gap mentioned before. It also presents a simulator, developed to clarify the operation of the model.

After presenting the model to a developers group, it was demonstrated that the model is understandable, implementable and adaptable to all kinds of users, with an acceptance close to 100%.

Introducción

Existe poco desarrollo de Sistemas Multi-Agente (MAS, MultiAgent Systems) en nuestro medio. Esto se debe principalmente a la complejidad de las metodologías de desarrollo y a los modelos componentes de un MAS.

Los MAS son sistemas que solucionan problemas en forma paralela, esto es, varios agentes que ejecutan las diferentes tareas en forma paralela o secuencial. Por ejemplo, para construir una casa, tenemos tareas como: nivelar terreno, levantar paredes, estucar, entre otras. Por lo tanto, los agentes podrán ir seleccionando y ejecutado las tareas hasta finalizar el proyecto. Se debe considerar que un agente es un proceso computacional autónomo con iniciativa y capacidad de reaccionar en su ambiente.

Para asegurar el correcto funcionamiento de un MAS, se necesita una buena comunicación entre los agentes (e.g., avisar que están ejecutando una tarea, que han finalizado una tarea).

Implementar un MAS requiere diferentes modelos (e.g., Agente, Tarea, Experiencia, entre otros) y uno de los más importantes es el Modelo de Comunicación. Este modelo es complejo ya que requiere la implementación de tres componentes: (i) un lenguaje común (entendible por todos los agentes del sistema), (ii) protocolos de transporte (medio de trasmisión de mensajes) y (iii) la interacción (cómo se comunican los agentes). Es por esta razón que se hace necesario contar con un modelo de comunicación más sencillo que facilite la implementación de un MAS.

El capítulo de resultados (capítulo único del trabajo) se ha dividido en dos títulos. El primer título presenta el modelo de comunicación para Sistemas Multiagente utilizando web services, detallando cada uno de sus componentes e interrelaciones. El segundo título presenta el diseño e implementación de un simulador que clarifica el funcionamiento del modelo de comunicaciones. Finalmente se presentan las conclusiones y sugerencias del trabajo.

Capítulo único: Resultados

El desarrollo de la tesis involucró dos actividades importantes: Diseñar el modelo de comunicación para sistemas multiagente utilizando web services y la implementación de un simulador que ayude a comprender la funcionalidad del modelo propuesto. Este capítulo se ha dividido en dos títulos, de acuerdo a las actividades mencionadas.

1. MODELO DE COMUNICACIÓN UTILIZANDO WEB SERVICES

El trabajo propuesto no pretende "inventar" un nuevo modelo de comunicación, lo que se pretende es adaptar los modelos de comunicación en un modelo que permita un claro entendimiento del mismo, así como una posterior implementación con tecnología actual y de fácil acceso.

Tomando como base los tres componentes básicos de la comunicación presentados en [VÁSQUEZ 2004] y [FININ, 1995], procederemos a enfocar la propuesta de la presente investigación:

- *Componente de comunicación*, componente que se constituye en el objeto de nuestro estudio.
- *Componente de representación*, encargado del entendimiento mutuo (semántica). Este componente está fuera del alcance del presente documento.
- *Componentes no relacionados con el conocimiento compartido*, no participan directamente de la comunicación, ayudan al mejor desempeño del agente. También fuera del alcance del estudio planteado.

En la Figura 1 se pueden apreciar estos tres componentes y la forma en la cual interactúan. En esta figura se resalta el componente que es objeto del presente documento: *Componente de Comunicación*. A continuación recordaremos los tres elementos constitutivos de este componente.

- *Protocolo de transporte*: Mecanismo de transporte usado para la comunicación.
- *Lenguaje de comunicación*: Medio por el cual las actitudes acerca del contenido del intercambio se comunican, es decir, poder saber si el contenido de la comunicación es una pregunta, una aseveración o una solicitud.

- *Protocolo de interacción*: Es la estrategia de alto nivel seguida por el agente de software que gobierna su interacción con otros agentes. Ese protocolo puede ir desde esquemas de negociación y de teoría de juegos protocolos de protocolos tan simple como “cada vez que usted no sabe algo, busque a alguien que sepa y pregunte".

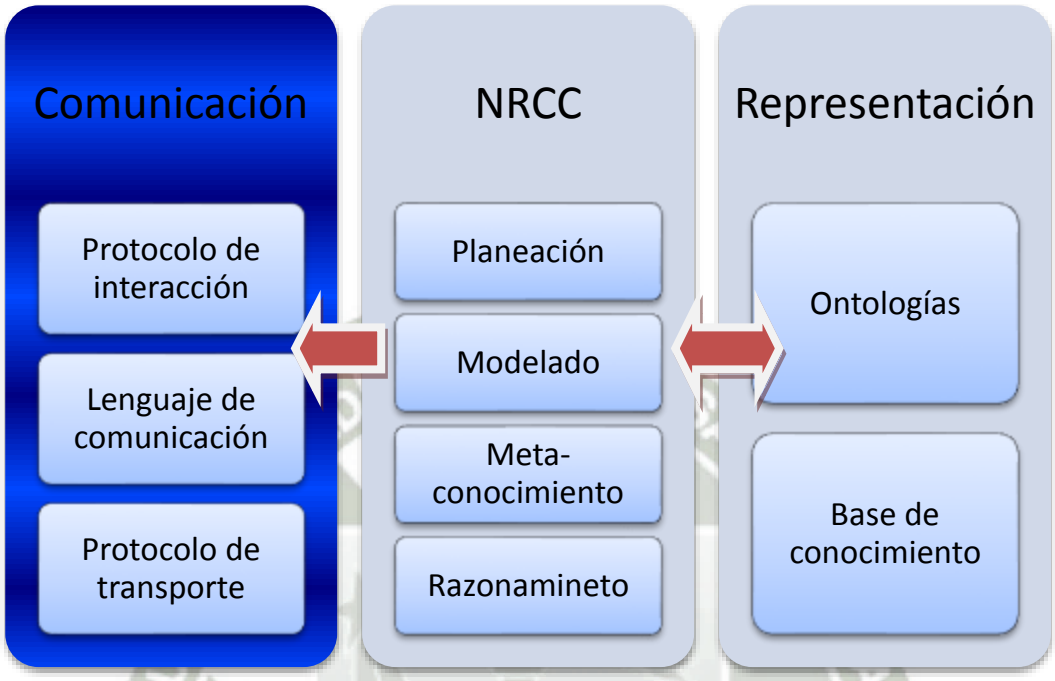


Figura 1: *Componentes básicos de la comunicación.*
Fuente: *Adaptación de propuesta de [Vásquez 2004] y [FININ, 1995].*

Es bueno aclarar en este punto, que muchas metodologías proponen un modelo de coordinación (e.g., [IGLESIAS, 1998]) en lugar de un modelo de comunicación. Intuitivamente se puede derivar que el modelo de coordinación es mucho más amplio y completo, y además no es parte del presente trabajo.

1.1. Modelo Propuesto

La Figura 2 resume el modelo propuesto en el presente trabajo. Se puede apreciar que de los tres componentes de un Modelo de Comunicación, dos serán propuestos y/o adaptados en este capítulo y el otro aprovechará las ventajas de la tecnología que se propone integrar al modelo.

La propuesta se resume de la siguiente manera:

- *Protocolos de transporte*: Se utilizarán los protocolos propios de los Web Services: HTTP sobre TCP y SOAP.
- *Protocolo de interacción*: La estrategia se basará en un modelo de pizarra, en combinación con un modelo de comunicación asistida, donde el facilitador es la pizarra. La estrategia se basa en la siguiente estructura básica:
  - ✓ Cada vez que un agente lleva a cabo una tarea informa el resultado a través de la pizarra.
  - ✓ Cada vez que un agente emprende una tarea, consulta a la pizarra el estado de dicha tarea.
- *Lenguaje de comunicación*: Se hará una adaptación del estándar KQML, para que el modelo no "pretenda" establecer un nuevo estándar. Esta adaptación utiliza las estructuras XML y HTML propias de los Web Services.

Figura 2: *Modelo de comunicación propuesto.*
Fuente: *Elaboración propia.*

De lo anterior se debe entender que la comunicación entre los agentes será indirecta (a través del facilitador, la pizarra), lo cual se representa en la figura 3.

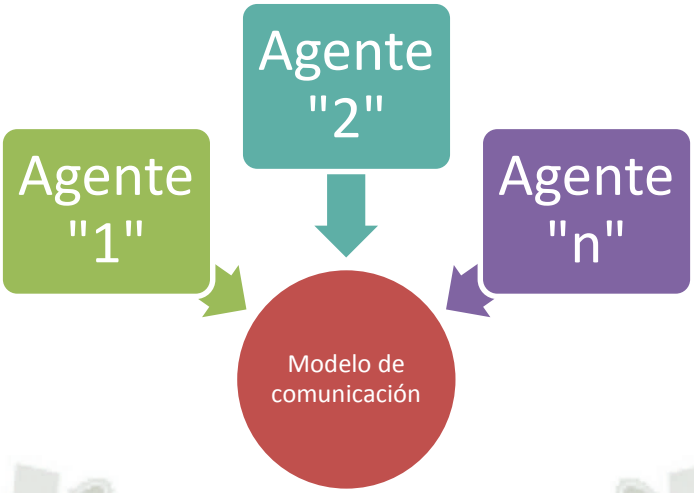


Figura 3: *Interacción asistida de los agentes.*
Fuente: *Elaboración propia.*

1.2. Diseño del Modelo

1.2.1. Protocolo de interacción

i. Objetivos

- Informar a los diferentes agentes del sistema de todo aquel conocimiento que se va adquiriendo.
- Dar acceso sin restricciones a los agentes del sistema, sobre las diferentes acciones que lleven a cabo los demás agentes.

ii. Elementos

Los elementos constituyentes del Protocolo de interacción del modelo de Comunicación que se está proponiendo son:

- Modelo asistido.
- Modelo de pizarra.

La figura 4 muestra la estructura del protocolo de interacción, destacando la relación entre ambos elementos (las clases que implementan estos elementos a través de web services). En la tabla 1 se describe las operaciones de ambos elementos.

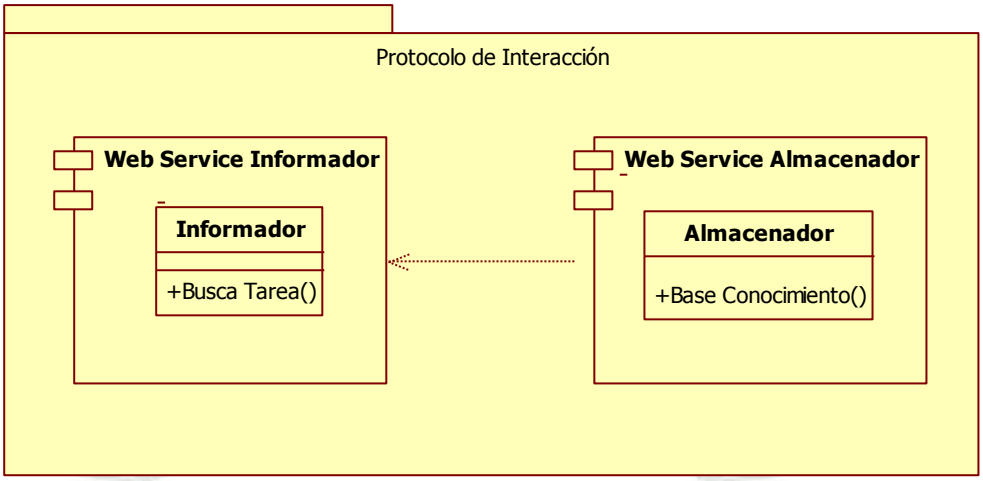


Figura 4: *Elementos constituyentes de Protocolo de interacción.*
Fuente: *Elaboración propia.*

Clase::Operación	Descripción
Informador::Busca_Tarea	Acepta la consulta de un agente y busca si existe alguna tarea disponible para ejecutar.
Almacenador::Base_Conocimiento	Se encarga de informar a cada agente del sistema las tareas que se están ejecutando.

Tabla 1: *Descripción de las clases del Protocolo de Interacción.*
Fuente: *Elaboración propia.*

iii. Funcionalidad

Los dos elementos mencionados en el punto anterior combinarán sus características para permitir la comunicación de los agentes. Para poder entender la funcionalidad, se hará una descripción por separado de cada uno de ellos.

- *Modelo asistido*: Este tipo de modelo permite la comunicación entre agentes a través de un facilitador, un ente que permita que esta comunicación sea clara y no lleve a confusiones. Para el modelo propuesto

se piensa utilizar una "pizarra" –que se describe líneas abajo – como mecanismo facilitador.

Los facilitadores se deben encargar de hacer llegar el nuevo conocimiento adquirido (e.g., una acción realizada, la toma de una tarea, entre otros) a los diferentes agentes, es en este punto donde tenemos un primer servicio web encargado de la función de informar.

- *Modelo de pizarra*: Este modelo es en esencia una estructura de datos persistente, cuyo objetivo es almacenar el conocimiento que está surgiendo en el sistema. Este almacenamiento se llevará a cabo a través de otro servicio web, que se gatillará cada vez que un agente cambie de estado.

  Se debe tener en consideración que la interpretación del conocimiento (cf. Figura 1, componente de representación) le corresponde a cada agente, del PAMA[1] que cada diseñador lleve a cabo al momento de diseñar los agentes, por lo tanto no es parte del protocolo de interacción y escapa del ámbito del presente trabajo de investigación.

La figura 5 describe el funcionamiento del protocolo en su conjunto. El cual comienza con alguna acción de los agentes (e.g., tomar una tarea para ejecutar, solicitar ayuda en una tarea, indicar que finalizó una tarea, entre otros), lo cual gatilla al servicio web "Almacenador". Este servicio web se encarga de escribir en la pizarra (estructura de datos) este nuevo conocimiento.

Una vez que la pizarra ha sido actualizada se dispara el servicio web "Informador", que se encarga de publicar el nuevo conocimiento a todos los agentes, esto es, les transmite este nuevo conocimiento de forma global.

[1] PAMA: **P**ercepciones, **A**cciones, **M**etas, **A**mbiente del agente.

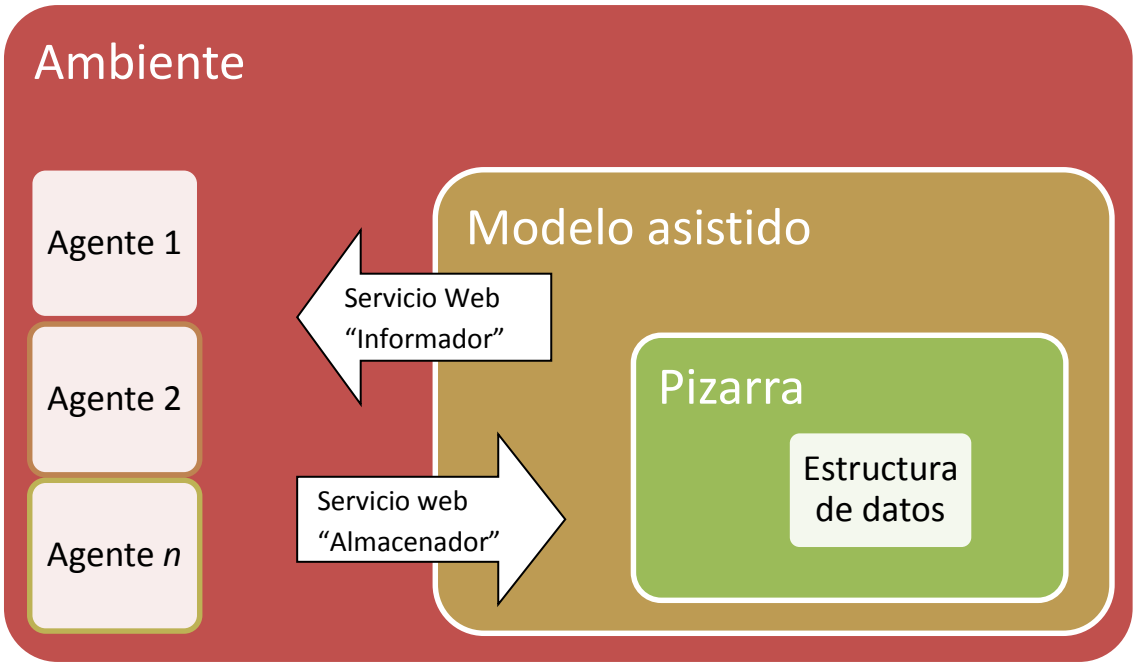


Figura 5: *Protocolo de interacción del Modelo de Comunicación propuesto.*
Fuente: *Elaboración propia.*

La estructura de la pizarra se define en el diagrama de clases de la figura 6.

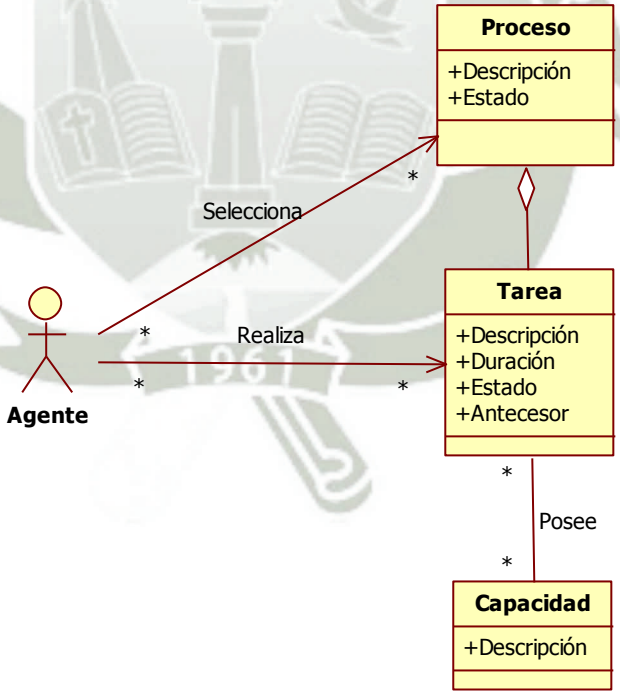


Figura 6: *Diagrama de clases de la pizarra.*
Fuente: *Elaboración propia.*

La pizarra está compuesta por 4 clases, las cuales se describen en la Tabla 2.

Clase	Descripción
Proceso	Define el o los procesos que se pueden resolver en el sistema multiagente (e.g., construir una casa)
Tarea	Todo proceso está compuesto de tareas, esta clase describe las componentes de los procesos.
Capacidad	Toda tarea requiere una cierta capacidad que poseen los agentes (i.e., no todos los agentes pueden hacer todas las tareas). Aquí se definen las capacidades de cada tarea.
Agente	Representa a los agentes que participan en el sistema.

Tabla 2: *Descripción de las componentes de la pizarra.*
Fuente: *Elaboración propia.*

La descripción de cada uno de los atributos de las clases que componen la pizarra se tiene en la tabla 3.

Clase	Atributo	Descripción
Proceso	Descripción	Definición del proceso
	Estado	Si el procesos está finalizado o no
Tarea	Descripción	Definición de la tarea
	Duración	Tiempo estimado de duración de la tarea
	Estado	Si la tarea está finalizada o no
	Antecesor	Define las tareas que deben estar finalizadas para que esta se pueda ejecutar
Capacidad	Descripción	Definición de la capacidad

Tabla 3: *Descripción de atributos de los componentes de la pizarra.*
Fuente: *Elaboración propia.*

iv. Interrelación

El protocolo de interacción se relaciona con los dos otros elementos del modelo de comunicación de la siguiente manera:

- *Lenguaje de comunicación*: El protocolo de interacción necesita conocer el lenguaje en el cual se elaboran los mensajes. Esto se da de manera transparente, debido a que los lenguajes son XML y HTML, propios de los servicios web utilizados en la interacción.
- *Protocolo de transporte*: Los servicios web –utilizados por el protocolo de interacción – se desplazan a través de los protocolos HTTP y SOAP, por lo que el viaje de los mensajes se da de manera transparente.

La figura 7 abstrae estas relaciones.

Figura 7: *Interrelaciones del protocolo de interacción.*
Fuente: *Elaboración propia.*

1.2.2. Lenguaje de comunicación

Como se mencionó, no se está creando un lenguaje de comunicación, se está adaptando uno existente –KQML – al utilizado por los web services: XML y HTML.

i. Objetivo

- Permitir la comunicación entre los diferentes agentes del sistema.

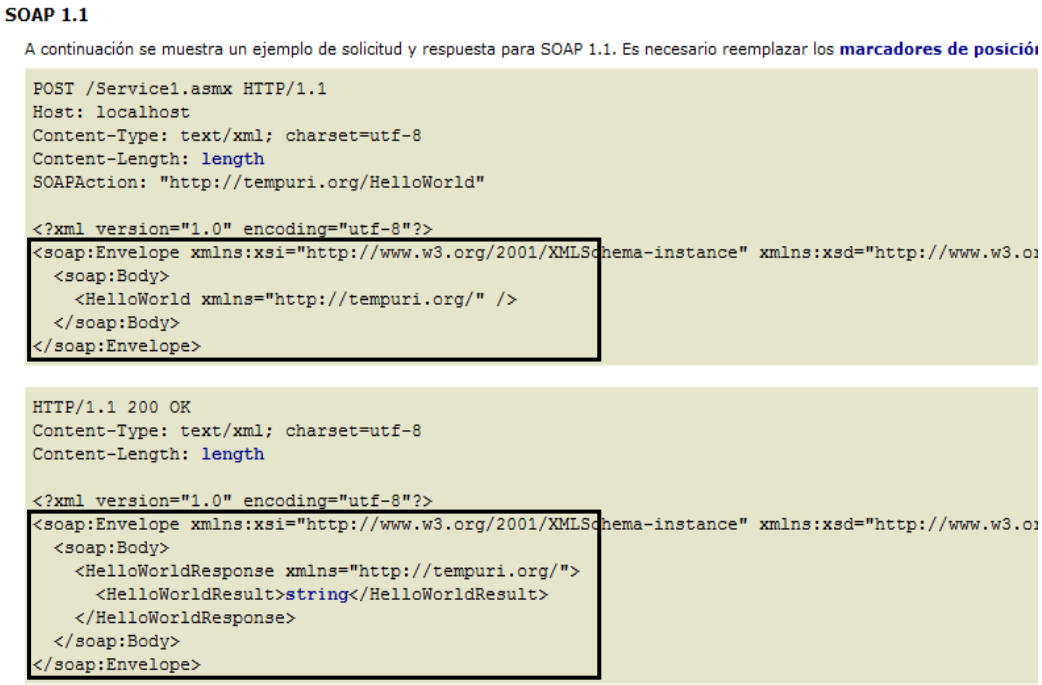
ii. Elementos

Los tres componentes de KQML son la capa de contenido (mensaje propiamente dicho), la capa de comunicación (define emisor, receptor e identificador único del mensaje), y por último, la capa de mensaje (relacionado a la codificación del lenguaje, protocolos y las preformativas).

Veamos a través de un ejemplo como es que estas tres capas se adaptan totalmente a los mecanismos utilizados por la tecnología de web services.

Al crear un proyecto de web services en Visual Studio, podemos obtener una pantalla como la que se muestra en la figura 8. Es un ejemplo de solicitud a un servicio web y la respuesta asociada. Veamos ahora las tres capas:

- *Capa de contenido*: Es el mensaje, que en la figura 8 se muestra en el recuadros (solicitud y respuesta).
- *Capa de comunicación*: Parte de esta capa es "invisible" en el código mostrado en la figura 8. Algunas partes se pueden apreciar: Host, Content-Type, Content-Length, etc.
- *Capa de mensaje*: Esta capa es todo el código mostrado en la figura 8, de aquí se deben resaltar dos elementos importantes:
  a. Preformativas: POST (consulta o petición) y HTTP (respuesta).
  b. Protocolo: SOAP



SOAP 1.1

A continuación se muestra un ejemplo de solicitud y respuesta para SOAP 1.1. Es necesario reemplazar los **marcadores de posició**

```
POST /Service1.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/HelloWorld"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.o
  <soap:Body>
    <HelloWorld xmlns="http://tempuri.org/" />
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.o
  <soap:Body>
    <HelloWorldResponse xmlns="http://tempuri.org/">
      <HelloWorldResult>string</HelloWorldResult>
    </HelloWorldResponse>
  </soap:Body>
</soap:Envelope>
```

Figura 8: *Solicitud y respuesta en SOAP.*
Fuente: *Microsoft Visual Studio.*

El mensaje SOAP viene empaquetado (SOAP Envelope) y está compuesto por dos partes: Cabecera y el Cuerpo. Esto lo podemos ver en la figura 9.

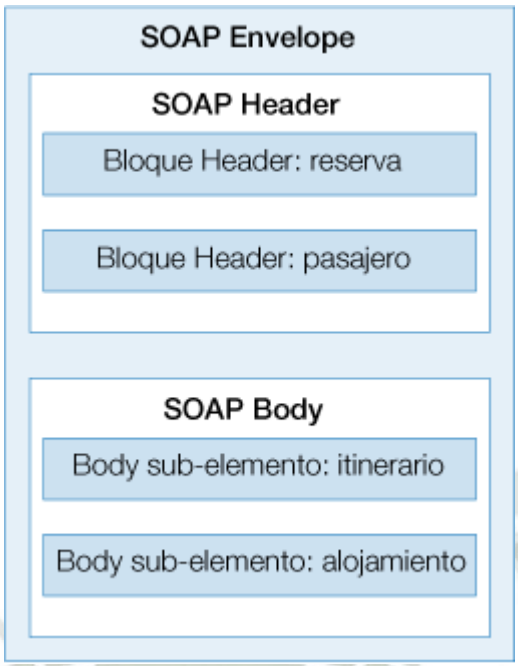


Figura 9: *Estructura de un mensaje SOAP.*
Fuente: *Cómo funciona un web service*[2].

Otro protocolo que se utiliza es HTTP POST, un ejemplo del código asociado a este protocolo se puede apreciar en la figura 10. En la figura se puede apreciar de una manera resumida las tres capas anteriormente mencionadas.

HTTP POST

A continuación se muestra un ejemplo de solicitud y respuesta para HTTP POST. Es necesario reemplazar los **m**

```
POST /Service1.asmx/HelloWorld HTTP/1.1
Host: localhost
Content-Type: application/x-www-form-urlencoded
Content-Length: length
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">string</string>
```

Figura 10: *Solicitud y respuesta en HTTP POST.*
Fuente: *Microsoft Visual Studio.*

El resultado de utilizar el protocolo HTTP POST, para el clásico ejemplo "*Hello World*" se muestra en la figura 11. Se puede apreciar que este resultado está codificado en XML.

[2] http://webservicecofu.blogspot.com/2007/06/como-funciona-el-web-service.html

iii. Funcionalidad

La funcionalidad está ligada a los web services, los cuales utilizan un formato conocido (XML) que se codifica por SOAP, transportan por TCP/IP y hacen transferencia a través de HTTP. La funcionalidad se verá con detalle en la siguiente sección (cf. 1.3.3 Protocolo de transporte).

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://tempuri.org/">Hello World</string>
```

Figura 11: *Respuesta del protocolo HTTP POST.*
Fuente: *Microsoft Visual Studio.*

iv. Interrelaciones

- La relación con el protocolo de interacción se da en la transmisión de los mensajes (mensaje codificado en XML por SOAP), este es el lenguaje.
- La relación con el protocolo de transporte es más fuerte, ya que el mensaje codificado necesita un mecanismo de transferencia (HTTP) y una vía de comunicación (TCP/IP).

1.2.3. Protocolo de transporte

Como se mencionó anteriormente, se utilizará el mecanismo de transporte de los web services.

i. Objetivos

- Permitir la transmisión de los mensajes entre los agentes y el modelo de comunicación.

ii. Elementos

Los elementos son los siguientes:

- SOAP: Es un protocolo que define el formato XML para los mensajes de intercambio en el uso de un Web Service.
- XML: Lenguaje utilizado por un Web Service para especificar de qué forma hay que proporcionarle los datos, de forma tal que cualquiera pueda interaccionar con el mismo.

- TCP/IP: Protocolos de transporte, sobre él viaja HTTP.
- HTTP: Protocolo de transferencia de hipertexto.

iii. Funcionalidad

Los siguientes pasos definen de manera simple y clara el funcionamiento de este componente:

- El Web Service empaqueta el mensaje en formato XML, utilizando el protocolo de comunicación SOAP.
- Este mensaje se envía al agente y/o pizarra a través del protocolo TCP/IP.
- El HTTP permite la transferencia en un entorno web, el que utilizan los web services. Este protocolo se complementa con TCP/IP.

iv. Interrelaciones

- La relación con el Lenguaje de comunicación se da a nivel de protocolos, las razones han sido ampliamente expuestas en esta sección.
- La relación con el protocolo de interacción es de ida y vuelta (cf., figura 7), ya que los mensajes viajan desde y hacia los agentes.

2. IMPLEMENTACIÓN DEL SIMULADOR

En este título se presentan los pasos para la implementación de un simulador que mostrará el funcionamiento del modelo de comunicación propuesto. La tabla 4 muestra las características del simulador.

Característica	Descripción
Ambiente general	Desarrollo de procesos
Procesos involucrados	• Elaboración de libros • Construcción de una casa
Restricciones	• Cada proceso tiene establecidas las precedencias entre las tareas que los componen. • Se supone que los agentes poseen las capacidades para desarrollar las tareas involucradas.
Agentes involucrados	3

Tabla 4: *Características generales del sistema multiagente simulado.*
Fuente: *Elaboración propia.*

Las tareas – y sus restricciones – de cada uno de los procesos involucrados se muestran en la tabla 5.

Elaboración de libros	Construcción de una casa
1. Definir estructura	1. Comprar terreno
2. Investigar capítulo I (Antecesor 1)	2. Elaborar planos
3. Escribir capítulo I (Antecesor 2)	3. Aprobar planos (Antecesor 2)
4. Investigar capítulo II (Antecesor 1)	4. Limpiar terreno (Antecesores 1, 3)
5. Escribir capítulo II (Antecesor 4)	5. Nivelar terreno (Antecesor 4)
6. Investigar capítulo III (Antecesor 1)	6. Excavar los cimientos (Antecesor 5)
7. Escribir capítulo III (Antecesor 6)	7. Vaciado de cimientos (Antecesor 6)
8. Editar texto (Antecesores 3,5,7)	8. Levantar paredes (Antecesor 7)
9. Organizar referencias (Antecesor 8)	9. Techar (Antecesor 8)
10. Escribir prólogo, resumen, índices, etc. (Antecesor 9)	10. Instalaciones sanitarias (Antecesor 9)
11. Imprimir (Antecesor 10)	11. Instalaciones eléctricas (Antecesor 9)
12. Enviar empastar (Antecesor 11)	12. Estucado (Antecesores 10, 11)
	13. Vaciado piso (Antecesor 12)
	14. Colocar piso (Antecesor 13)
	15. Colocar mayólicas (Antecesor 14)
	16. Colocar puertas (Antecesor 14)
	17. Colocar ventanas (Antecesor 14)
	18. Pintar (Antecesores 15, 16, 17)

Tabla 5: *Tareas y restricciones de los procesos del sistema multiagente simulado.*
Fuente: *Elaboración propia.*

2.1. Análisis

Los requerimientos del simulador son cuatro, y se listan a continuación (fig. 12).

1. El sistema multiagente define uno de los procesos involucrados como activo (e.g., Elaboración de libros). Una vez activo el proceso, los agentes comenzarán a consultar por actividades libres. Para esto se llama al Servicio Web Informador, el cual consulta a la Pizarra por las actividades libres. Cuando se encuentre una actividad libre se asigna al Agente que realiza la consulta.
2. Luego de asignar la actividad, se debe cambiar su estado y se debe informar a todos los agentes que dicha tarea está siendo ejecutada por algún Agente (este conocimiento lo procesa el agente y lo almacena en su base de conocimientos). Esta etapa la lleva a cabo el Servicio Web Almacenador.
3. Cuando no se encuentre ninguna actividad libre, se debe informar a los agentes para que no realicen más consultas y finalice el proceso.
4. Para los agentes es transparente el proceso a llevar a cabo, debemos recordar que es un simulador y la cantidad de procesos fue elegida para tener una mejor representación del modelo de comunicaciones y hacerlo más atractivo al usuario.

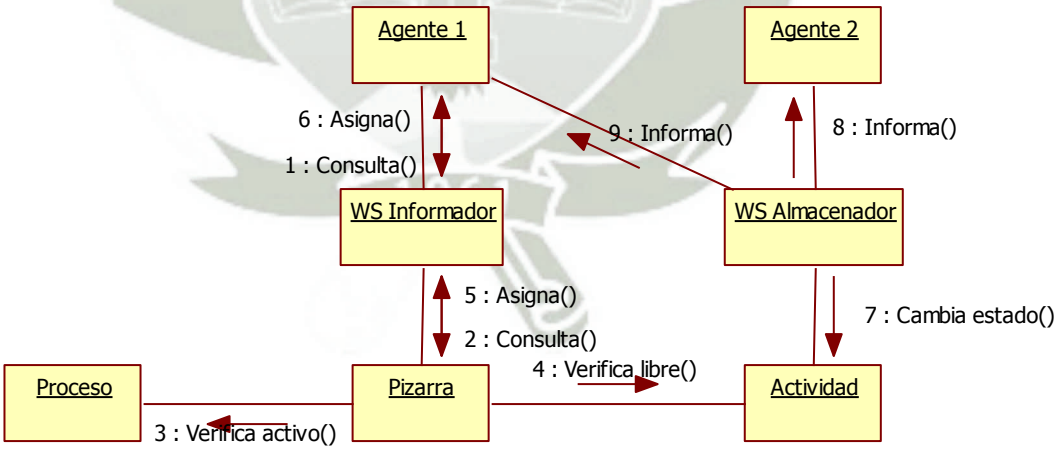


Figura 12: *Diagrama interacción del simulador.*
Fuente: *Elaboración propia.*

2.2. Diseño

2.2.1. Diagrama de clases

El diagrama de clases del simulador se muestra en la figura 13. En este diagrama se pueden identificar tres capas importantes:

- *Capa de interfaz:* Compuesta por las clases Agente e Interfaz Simulador. La implementación de estas dos clases se hará en un solo módulo o proyecto y constituye toda la actividad simulada del sistema, i.e., actividad del agente, componente de representación y componentes no relacionados con el conocimiento compartido.
- *Capa de negocio:* Compuesta por las clases ws_Informador y ws_Almacenador, las cuales representan la parte funcional del Modelo de Comunicación, y se implementará a través de dos Servicios Web.
- *Capa de Almacenamiento:* Compuesta por la clase pizarra, la cual es una base de datos. Representa la parte de almacenamiento del Modelo de Comunicación.

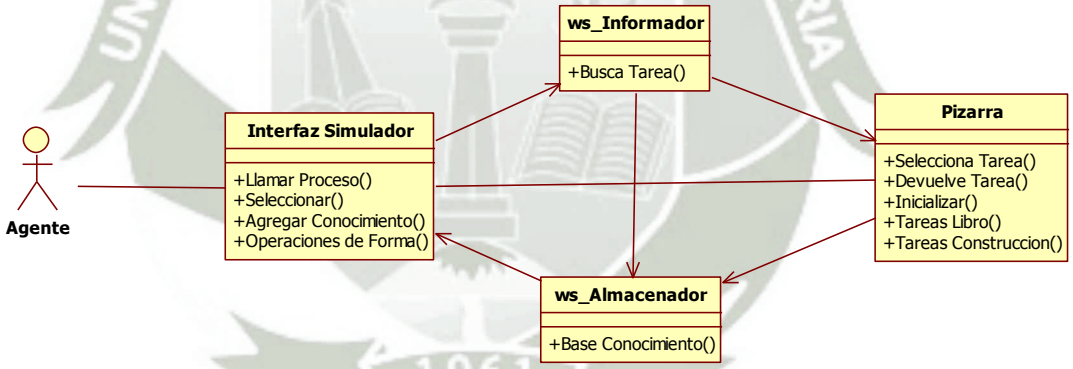


Figura 13: *Diagrama de clases del simulador.*
Fuente: *Elaboración propia.*

En la tabla 6 se describen las operaciones de las clases del simulador del Modelo de Comunicaciones.

2.2.2. Servicios web

Ambos servicios web se implementarán en VB.NET. Los códigos de las clases y los métodos se encuentran en el Anexo 4.

Clase	Atributo	Descripción
Interfaz Simulador	Llamar Proceso	Se comunica con la base de datos para obtener las tareas de los procesos y cargarlas en una grilla.
	Seleccionar	Operación principal, simula las consultas de los agentes y se comunica con los Servicios Web, para buscar tareas libres (llama la operación Busca Tarea de ws_Informador) y obtener dichas tareas (llama la operación Base Conocimientos de ws_Almacenador).
	Agregar Conocimiento	Simula el procesamiento del conocimiento por parte de los agentes, esto es la recepción y almacenamiento del conocimiento por parte de los agentes.
	Operaciones de Forma	Las demás operaciones del simulador que se encargan de dar formatos, presentaciones visuales, etc.
ws_Informador	Busca Tarea	Se comunica con la base de datos para saber si existe alguna tarea libre en el proceso activo.
ws_Almacenador	Base Conocimientos	Se comunica con la base de datos para poder asignar una tarea a un agente.
Pizarra	Selecciona Tarea	Busca tareas libres en el proceso activo. Devuelve el ID de la tarea ó 0 cuando el proceso ha finalizado. Actualiza el estado de la tarea seleccionada y almacena la relación Agente-Tarea.
	Devuelve Tarea	Devuelve la última tarea seleccionada con el fin que se pueda informar a los agentes este conocimiento.
	Inicializar	Inicializa la base de datos para una nueva ejecución.
	Tareas Libro	Devuelve todas las tareas del proceso Elaboración de Libro.
	Tareas Construcción	Devuelve todas las tareas del proceso Construcción de una casa.

Tabla 6: *Descripción de atributos de las clases.*
Fuente: *Elaboración propia.*

2.2.3. Pizarra

Como ya se mencionó líneas arriba, la Pizarra es una base de datos. El modelo Entidad Relación se muestra en la figura 14.

Las cinco operaciones de la pizarra han sido implementadas a través de procedimientos almacenados: *sp_Selecciona_Tarea*, *sp_Devuelve_Tarea*, *sp_Inicializar*, *sp_TareasLibro_Tarea*, *sp_TareasCasa_Tarea*. El código de cada uno de los procedimientos se encuentra en el Anexo 4.

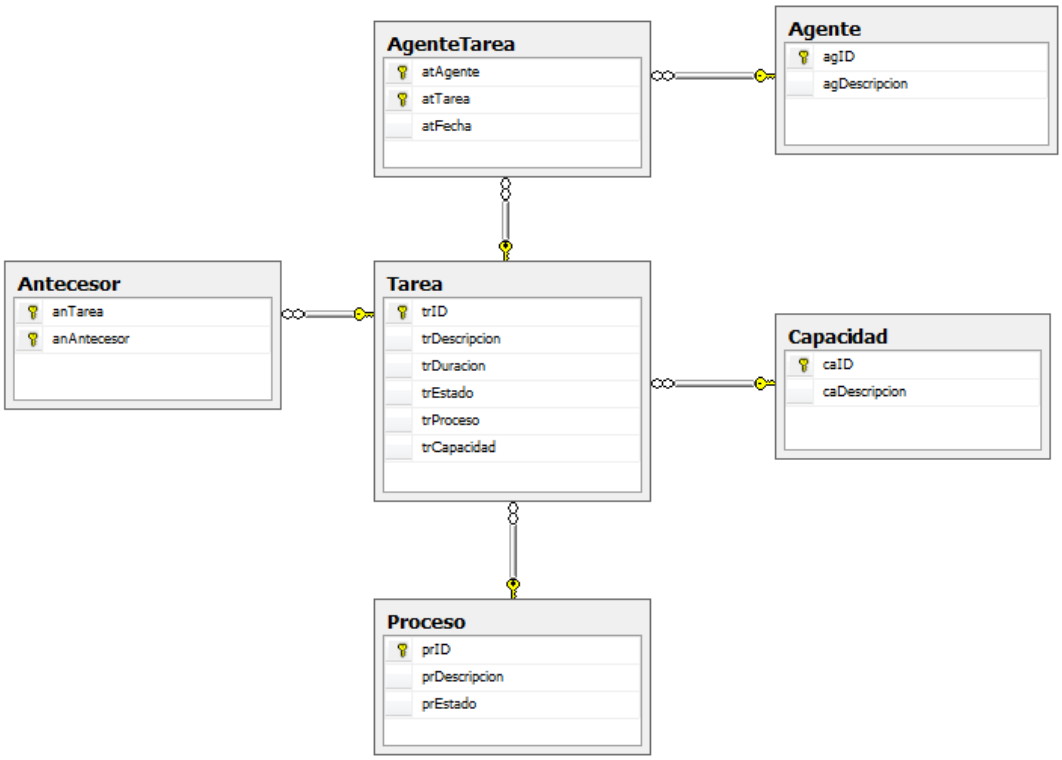


Figura 14: *Modelo Entidad-Relación de la Base de Datos de la Pizarra.*
Fuente: *Elaboración propia.*

2.2.4. Ambiente de simulación

La funcionalidad del ambiente de simulación (código de las funciones más importantes) se encuentra en el Anexo 4. En las figuras 15 y 16 se muestran la pantalla principal y la pantalla de solución del simulador, respectivamente. En el Anexo 3 se encuentran todas las pantallas restantes.

La figura 15 muestra la interfaz gráfica principal del simulador. En ella podemos apreciar el proceso de Elaboración de un libro en funcionamiento. Se puede observar que los tres agentes están actualizando –acumulando- su conocimiento, también se puede apreciar que el *Agente 2* está activo y llevando a cabo la actividad *Investigar capítulo I.* En el lado derecho se observa el estado y seguimiento de las actividades.

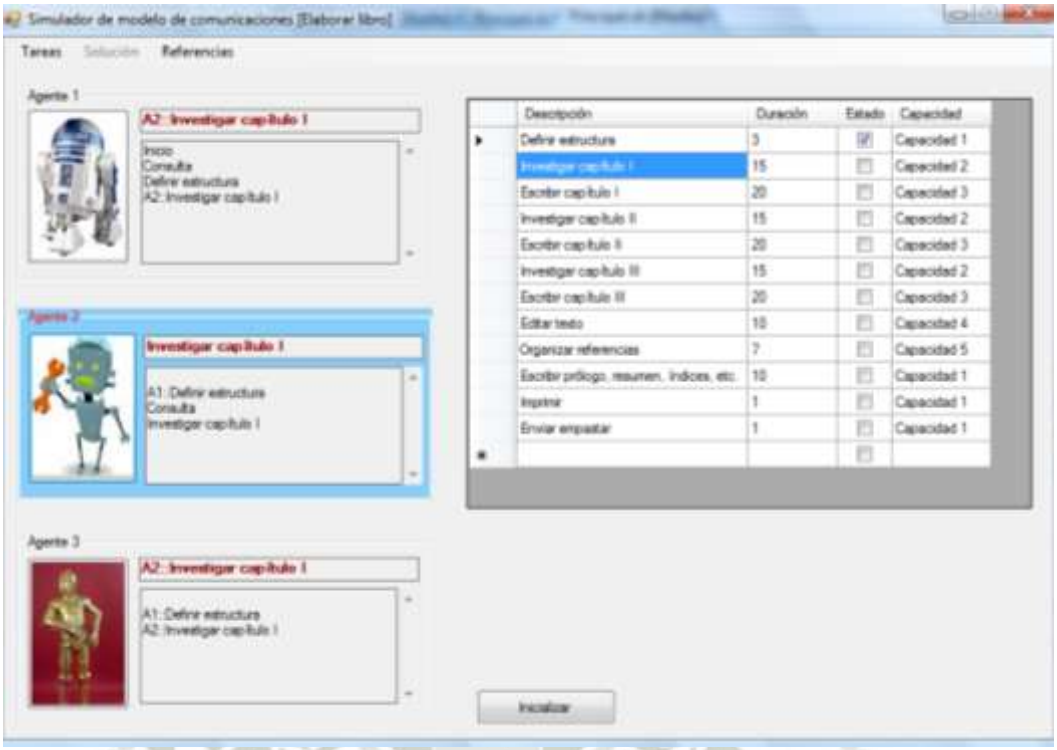


Figura 15: *GUI principal del simulador.*
Fuente: *Elaboración propia.*

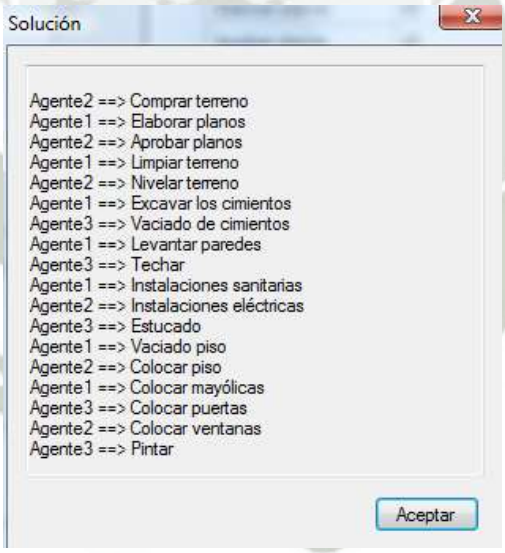


Figura 16: *GUI que muestra una solución al proceso de construcción de una casa.*
Fuente: *Elaboración propia.*

La figura 16 muestra el resultado del proceso de construcción de una casa. Se puede apreciar el orden en que fueron ejecutadas las actividades, así como los agentes que las llevaron a cabo.

2.3. Implementación

Las características técnicas de la implementación se muestran en la tabla 7.

Característica	Descripción
Sistema Operativo	Windows Vista Business
Entorno de desarrollo	Visual Studio Team System 2008
Lenguaje de programación	Visual Basic 2008
Gestor de base de datos	SQL Server 2008
Servidor Web	Internet Information Services (IIS) 7
Código de implementación	El código de implementación se encuentra en el Anexo 4.

Tabla 7: *Características técnicas del Simulador de Modelo de Comunicación.*
Fuente: *Elaboración propia.*

2.4. Funcionamiento

En el Anexo 3 se muestra una guía de funcionamiento del simulador.

IMPORTANTE: No se ha profundizado en algunos tópicos del diseño del simulador, debido a que sería repetir en gran parte el Título 1 (modelo de comunicación utilizando web services)

Conclusiones

PRIMERA

Se ha propuesto un Modelo de Comunicación para Sistemas Multiagente haciendo uso de la tecnología de Servicios Web. Este modelo es entendible, implementable y adaptable a todo tipo de usuario.

SEGUNDA

La estructuración del Modelo de Comunicación que se ha propuesto, hace que sea sencillo de implementar y de utilizar.

TERCERA

Se llevó a cabo la simulación del Modelo de Comunicación, la cual permitió entender mejor el modelo y comprobar su funcionamiento. Para esto se desarrolló un simulador que trabaja con tres agentes y dos procesos.

CUARTA

El Modelo de Comunicación para Sistemas Multiagente utilizando Web Services es entendible. Esto lo corrobora el 88% de los encuestados, luego de leerlo. También, el 94% de los encuestados lo entendieron aun mejor al utilizar el simulador presentado.

QUINTA

El Modelo de Comunicación para Sistemas Multiagente utilizando Web Services es implementable. Esto lo corrobora el 81% de los encuestados, quienes lo encuentran muy sencillo de implementar. Se debe agregar que el 100% de los encuetados considera que además de sencillo complementa muy bien a los Sistemas multiagente.

SEXTA

El Modelo de Comunicación para Sistemas Multiagente utilizando Web Services es adaptable. Esto lo corrobora el 100% de los encuestados, quienes consideran que el nivel de conocimiento de los usuarios no será una barrera. Se debe agregar que el 75% considera que la adaptación será muy fácil.

De esta manera se lograron los objetivos trazados en el presente trabajo y la hipótesis fue comprobada.

Sugerencias

PRIMERA

El Modelo de Comunicación propuesto en el presente trabajo se debería difundir y utilizar en el Programa Profesional de Ingeniería de Sistemas, específicamente en la línea de especialización de Inteligencia Artificial. Esta difusión debería servir para elevar el nivel de comprensión en esta línea de investigación e incrementar el desarrollo de los MAS.

SEGUNDA

La difusión del Modelo de Comunicación propuesto debe considerar el Simulador desarrollado en el trabajo, debido a que éste ayuda a entender mejor el modelo y su funcionamiento en un sistema multiagente real.

TERCERA

Se debe continuar el estudio buscando nuevas tecnologías que hagan más asequibles los MAS, y complementar los modelos no vistos en este trabajo (e.g., Componente de Representación).

Propuesta de desarrollo

Título de la Propuesta

Implementación del Modelo de Comunicación para Sistemas Multiagente utilizando Web Services en cursos de Inteligencia Artificial del Programa Profesional de Ingeniería de Sistemas de la Universidad Católica de Santa María.

Justificación

Los sistemas computacionales requieren de mejores métodos para solucionar problemas complejos que surgen día a día, uno de estos métodos son los sistemas multiagente. Actualmente no existe un desarrollo masivo de este tipo de sistemas, una de las razones es su poca divulgación y complejidad de los modelos necesarios para su implementación. Entre los modelos necesarios para implementar un sistema multiagente se tiene: Modelo de agente, modelo de tareas, modelo de coordinación y modelo de comunicación. Este último modelo es uno de los más importantes y también uno de los más complejos de implementar.

Por lo tanto, divulgar el modelo propuesto puede ayudar a que se incremente el desarrollo de este tipo de sistemas y por lo tanto contar con métodos que permitirán solucionar diversos tipos de problemas.

El Programa Profesional de Ingeniería de Sistemas desarrolla tres líneas de especialización. Una de estas líneas es Inteligencia Artificial, donde se llevan a cabo cinco cursos de esta especialidad:

- Introducción a la Inteligencia Artificial
- Programación para Inteligencia Artificial
- Métodos de solución de problemas de Inteligencia Artificial
- Sistemas basados en el conocimiento
- Sistemas adaptativos

En los tres primeros cursos se desarrollan temas relacionados a agentes, por lo tanto introducir los sistemas multiagente en ellos no afectará la currícula. Todo lo contrario,

introducir este tema permitirá mantener una currícula actualizada y con temas novedosos.

Objetivos

- Diseñar el temario sobre Sistemas Multiagente que se introducirá en los cursos de Inteligencia Artificial.
- Diseñar un proyecto de desarrollo que involucre Sistemas Multiagente en los cursos de Inteligencia Artificial.
- Asegurar el uso del Modelo de Comunicación de Sistemas Multiagente utilizando Web Services en el temario y proyecto.

Recursos necesarios

Para la correcta ejecución de la propuesta se necesita:

- La participación de los profesores de la línea de Inteligencia Artificial del Programa Profesional de Ingeniería de Sistemas.
- Autorización del Programa Profesional de Ingeniería de Sistemas para poder modificar el contenido de los cursos.
- Un computador con acceso a Internet.
- Acceso a bibliografía sobre el tema.
- Un lugar físico para trabajar.

Metodología

Los pasos necesarios para desarrollar la propuesta son los siguientes:

1. Conocer el Modelo de Comunicación para Sistemas Multiagente utilizando Web Services.

   Se deberá capacitar a los profesores de la línea de Inteligencia Artificial del Programa Profesional de Ingeniería de Sistemas en el conocimiento y utilización del modelo de comunicación.

   Esta capacitación deberá estar dada por el autor del modelo y se deberá tener acceso al simulador que es parte del trabajo de investigación.

2. Selección de temario para introducir en cursos.

   En función al conocimiento del modelo de comunicación, se elegirán los temas relevantes y necesarios para asegurar la correcta implementación y utilización del modelo. Para esto, se necesita tener acceso a bibliografía reciente sobre el tema.

   Se deberá estructurar el temario de manera que se pueda introducir en tres cursos.

3. Diseño de proyecto de desarrollo que involucre el modelo de comunicación.

   Se diseña un proyecto que involucre el desarrollo de un Sistema Multiagente completo, y que involucre el Modelo de Comunicación propuesto.

   Este proyecto deberá considerar tres grandes etapas, cada una de ellas se desarrollará en cada uno de los cursos involucrados.

4. Elevar modificaciones a las autoridades correspondientes.

   Una vez que se tenga el temario, estructura de cursos y proyecto a desarrollar, se elaborará una propuesta a las autoridades universitarias respectivas para su correspondiente aprobación.

5. Implementar la propuesta

   Una vez aprobada la propuesta se ejecuta.

Cronograma de trabajo

La propuesta se deberá desarrollar en un máximo de 5 semanas, el detalle se muestra en la siguiente Carta Gantt.

Actividad	Duración (días)	Recursos
Conocimiento del modelo	3	Profesores, Aula, Computador
Diseño del Temario	10	Profesores, Computador, Bibliografía
Diseño de proyecto de desarrollo	5	Profesores, Computador, Bibliografía
Elevar propuesta para aprobación	5	Profesores, Computador
Ejecución	5	Profesores

Marco teórico

El conocimiento necesario para desarrollar la propuesta debe considerar los siguientes tópicos:

- Agentes.
- Sistemas Multiagente.
- Web Services.
- Modelos de Sistemas Multiagente.
- Modelos de Comunicación para Sistemas Multiagente.
- Modelo de Comunicación para Sistemas Multiagente utilizando Web Services.
- Aplicaciones de Sistemas Multiagente.

Resultados esperados

Lo que se espera como resultado de esta propuesta es:

- Contar con profesionales capacitados en el desarrollo e implementación de Sistemas Multiagente.
- La difusión de este tipo de sistemas para solucionar problemas complejos.
- Facilitar la implementación de este tipo de sistemas, utilizando un modelo claro y sencillo.

Bibliografía

[CARVER, 1992] Carver, N. – Lesser, V., "The evolution of blackboard control architectures", Technical report, University of Massachusetts Amherst, Oct. 1992.

[CORKILL, 1991] Corkill, D., "Blackboard systems", AI Expert, 6(9):40–47, Sept. 1991.

[FININ, 1995] Finin, T. – Labrou, Y. – Mayfield, J., "KQML as an agent communication language", Universidad de Ciencias de la Computación e Ingeniería Eléctrica, USA, 1995.

[GEORGEFF, 1989] Georgeff, M., "Communication and interaction in multi-agent planning", In L. Gasser and M. N. Huhns, editors, Distributed Artificial Intelligence. Volume II, pages 200–209,USA, 1989.

[IGLESIAS, 1998] Iglesias, C., "Definición de una Metodología para el Desarrollo de Sistemas Multiagente", Universidad Politécnica de Madrid, España, 1998.

[NEWELL, 1982] Newell, A., "The knowledge level", Artificial Intelligence, 18:87–127, 1982.

[ROSENSCHEIN, 1988] Rosenschein, J. – Genesereth, M., "Deals among rational agents", In A. H. Bond and L. Gasser, editors, Readings in Distributed Artificial Intelligence, pages 227–234,USA, 1988.

[VAZQUEZ, 2004] Vazquez, A., "Knowledge en Multiagent Systems", IJCAI 3.0 (Internet Junkie's Computer Artificial Intelligence), http://www.ijcai-03.org/~avazquez/knowledge/, 2004

[WERNER, 1989] Werner E., "Cooperating agents: A unified theory of communication and social structure", In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence Volume II, pages 3–36, USA, 1989.

[WOOLDRIDGE, 1995] Wooldridge, M. – Jennings, N., "Intelligent agents: Theory and practice". The Knowledge Engineering Review, 10(2):115–152, 1995.

Anexo 1: Proyecto de Tesis

I. PREÁMBULO

Encontrar personas que desarrollen sistemas multiagente en nuestro medio es muy complicado, por la necesidad de nuevo conocimiento que abarca. Las pocas personas que realizan este tipo de actividad lo hacen por investigación o por necesidad propia; algo en lo que coinciden ambos grupos de personas es que las diferentes metodologías existentes plantean una serie de modelos (simples y complejos) y además exigen aprender nuevas técnicas y plataformas que a la larga hacen desistir de entrar en este apasionante mundo.

Por los años de experiencia en el área de la Inteligencia Artificial he podido observar que es cierta la apreciación de estas personas y podría agregar que las técnicas y plataformas, en muchos casos, no se adaptan a novatos en la materia y quizás por esta razón adicional no se desarrollen este tipo de sistemas.

Por las razones planteadas me he decidido a plantear este trabajo de investigación ya que considero que el campo de la Inteligencia Artificial, específicamente en la línea de los sistemas multiagente, se encuentra mermado y necesita desarrollarse en nuestro medio.

II. PLANTEAMIENTO TEÓRICO

1. PROBLEMA DE INVESTIGACIÓN

1.1. Enunciado del problema

Modelo de comunicación para Sistemas Multiagente utilizando Web Services.

1.2. Descripción del problema

Los sistemas multiagente han cobrado una gran importancia en el mundo, debido a sus grandes ventajas respecto a agentes simples y a los sistemas de software comunes. Lamentablemente este tipo de sistemas no se desarrollan en nuestro medio, hablemos de nuestra región y de nuestro país.

Una de las principales razones por las cuales no se hacen este tipo de sistemas es que los modelos que las metodologías plantean no se adaptan o simplemente no

son claras para todos los usuarios. Esto genera la necesidad de simplificar estos modelos de tal forma que esté al alcance de todo tipo de usuario.

Las metodologías plantean modelos, y entre éstos tenemos el modelo de comunicación, que por su naturaleza y complejidad subyacente, ocasiona que muchos desistan en utilizar esta nueva tecnología. El problema no es tanto entender estos modelos, sino su implementación. Por lo tanto se necesita un modelo de comunicación que sea claro y además facilite la implementación con tecnología de fácil alcance y actual.

El proyecto se realizará en el área de las *Ciencias Formales*, y en la línea de *Ingeniería de Sistemas*.

Las variables del problema son el *Modelo de Comunicación para Sistemas Multiagente* debido a lo complejo de su implementación y sus indicadores son que sea *implementable* y que se *adapte a todo nivel*. La segunda variable es la *Utilización de los Web Services* para mejorar la complejidad planteada y su indicador es que sea *claro*.

El proyecto es de tipo *aplicado* y el nivel del mismo es *exploratorio*.

1.3. Justificación del problema

El desarrollo de software y su exportación se han convertido en armas muy poderosas que están siendo utilizadas por los países para mejorar su situación económica y hasta para intentar salir del subdesarrollo. Para que esta alternativa se pueda convertir en una realidad se necesita software de calidad que incluya técnicas avanzadas de desarrollo. El software empresarial moderno así como el software genérico de aplicación, ya no son simples programas de transacción de datos sino que son sistemas complejos que ayudan y soportan la toma de decisiones. Estos sistemas modernos utilizan técnicas de Inteligencia Artificial, entre ellas la de Sistemas Multiagente.

En nuestro medio no está muy difundido el desarrollo de sistemas multiagente, por las razones que ya fueron expuestas. Es por eso que mediante este trabajo se tratará de ampliar esta línea de la Inteligencia Artificial.

La justificación académica está dada por los alumnos del Programa Profesional de Ingeniería de Sistemas de nuestra Universidad, ya que ellos cuentan con una línea de especialización en Ingeniería del Conocimiento y por lo tanto este trabajo les será de mucha utilidad.

Las personas que conocen de un tema y trabajan con una metodología e incursionan en el tema de Sistemas Multiagente no tendrían problemas al momento de adoptar el modelo propuesto, pero para una persona novata en la materia esta tarea no es tan sencilla y muchas veces es mucho más complicada que desarrollar el sistema en sí. El modelo que se piensa proponer tratará de solucionar este problema y se adaptará a todo nivel de conocimiento de las personas sobre esta materia.

El presente proyecto es factible, ya que se cuenta con todos los recursos necesarios como son contactos, computadora, Internet y asesoría. Con estos recursos, además del tiempo, se podrá resolver el problema planteado.

Personalmente, considero que ya se debe comenzar a desarrollar de una manera diferente en nuestro medio, no debemos entramparnos en modelos complejos. Los sistemas multiagente son una solución bastante cercana a problemas que se viven todos los días en las empresas. Por lo tanto considero de mucha importancia la propuesta que planteo.

2. MARCO CONCEPTUAL

2.1. Agentes

Un agente según [RUSSSEL, 1996] "*es todo aquello que puede considerarse que percibe su ambiente mediante sensores y que responde o actúa en tal ambiente por medio de efectores*". De este concepto debemos sacar tres conceptos adicionales: Ambiente, Sensor y Efector, los cuales se proponen a continuación. La figura 01, resume el concepto de agente.

El ambiente es el medio en el cual se desenvuelve el agente, este puede ser accesible y/o determinista y/o episódico y/o estático y/o discreto o todo lo contrario. Para entender mejor los diferentes ambientes, veamos los conceptos que plantea [RUSSSEL, 1996]:

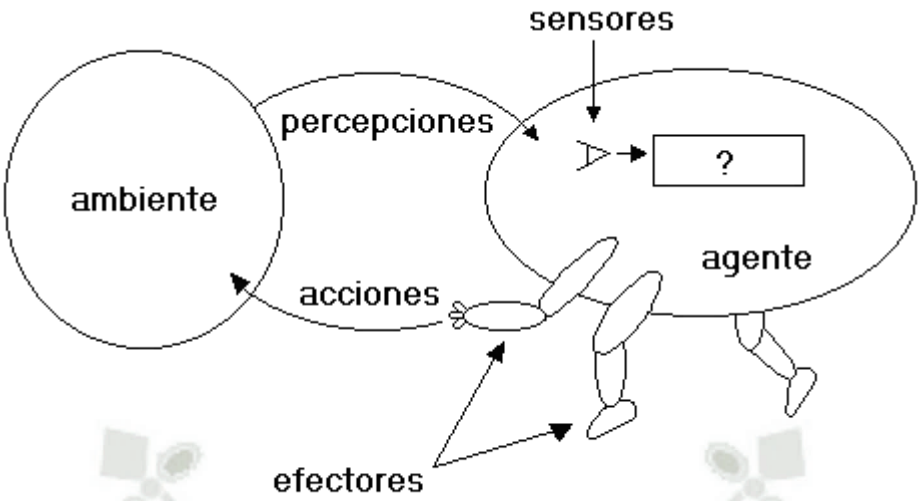


Figura 17: *Agente inteligente.*

Fuente: *[Russell, 1996].*

- *Ambiente accesible*: Es aquel ambiente al cual, el agente, tiene acceso total – a través de sus sensores – solo lo relevante. Un ambiente inaccesible es lo contrario.

- *Ambiente determinista*: Es aquel ambiente en el cual el agente puede determinar el siguiente estado o situación por: (i) el estado actual (situación actual o situación 0) y (ii) las acciones escogidas por el mismo agente u otros.

- *Ambiente episódico*: Es aquel ambiente en el cual, cada estado o situación en la que se encuentre un agente es independiente de otro estado. En este tipo de ambientes, los agentes no necesitan pensar por adelantado, solo se dedican a la situación actual.

- *Ambiente estático*: Es aquel ambiente que no sufre modificaciones mientras el agente delibera por la acción para llevar a cabo. El ambiente dinámico es aquel que varía con el tiempo y puede influir en la decisión tomada por el agente.

- *Ambiente discreto*: Es aquel ambiente del cual hay poco por percibir y poco por hacer, no se quiere decir que sea limitado, sino que las percepciones para el agente son pocas y sus acciones también son pocas. En cambio un

ambiente continuo es el que tiene ilimitadas percepciones y las acciones están determinadas por las percepciones y capacidades del agente.

Un sensor es un medio por el cual el agente percibe del ambiente, podemos tener sensores mecánicos, orgánicos o lógicos (software). Como ejemplos de sensores tenemos: Cámaras (visión), micros (oídos) y Velocímetro.

Un efector es el medio por el cual el agente responde a las percepciones, éstos pueden ser mecánicos, orgánicos o lógicos (software). Como ejemplos de efectores tenemos: Piernas mecánicas, brazos mecánicos o cualquier artefacto mecánico que pueda ser accionado por un agente.

Los agentes son una alternativa automática de utilizar los diversos sistemas con los que nos comunicamos a diario, esta alternativa plantea nuevos retos que debemos estar en capacidad de poder aceptar. Muchas veces, el hecho de escuchar el nombre "agente" surge un temor en los desarrolladores de software, cuando los agentes no son nada más que una nueva forma de programar soluciones de software.

Algo más, que se debe tomar en consideración en el tema de agentes, es su diseño. Como [RUSSSEL, 1996] plantea, su diseño debe tener en consideración al menos 4 aspectos: Percepciones, Acciones, Meta(s) y Ambiente. El autor lo plantea como un acrónimo el PAMA para diseñar agentes. Personalmente considero que en el PAMA también debe considerarse los sensores y los efectores que el agente podría tener.

2.2. Sistemas multiagente

Años atrás hablar de sistemas autónomos causaba algo de incertidumbre y temor entre las personas. Sin embargo, a la fecha se ha logrado grandes avances dentro de la Inteligencia Artificial y gracias a ello los investigadores intentan hallar cuáles son las implicaciones en el mundo real de múltiples "agentes" autónomos.

Uno de los grandes logros de la Inteligencia Artificial es el haber llegado a los Sistemas Distribuidos y con ello a los Sistemas Multiagente, aquellos sistemas que para llegar a un fin lo hacen a través de una búsqueda paralela entre sus

diversos agentes, quienes se dividirán el trabajo para que una vez finalizado estos resultados se unan llegando a una solución óptima y fidedigna.

La Inteligencia Artificial Distribuida (DAI, Distributed Artificial Intelligence), según [STONE, 1997] es un sub-campo de la Inteligencia Artificial que se preocupa de los sistemas consistentes en múltiples entidades independientes que interactúan en un dominio, es decir, dedicados al estudio de las técnicas y el conocimiento necesario para la coordinación y distribución del conocimiento y las acciones en un entorno con múltiples agentes. Como podemos apreciar en la figura 02, la DAI se ha dividido en dos áreas: Resolución de Problemas Distribuidos (DPS, Distributed Problem Solving) y Sistemas Multiagente (MAS, MultiAgent Systems).

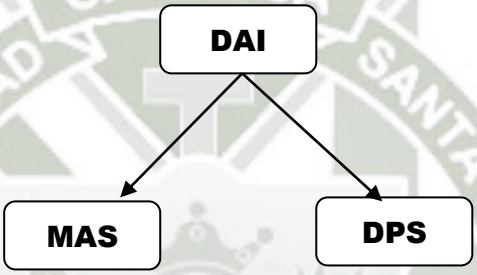


Figura 18: *Áreas de la Inteligencia Artificial Distribuida.*
Fuente: *Elaboración Propia.*

Las tareas de la DPS son de descomposición de un problema en varios sub-problemas - un número de módulos (o nodos) – que cooperan en dividir y compartir conocimiento acerca del problema y su correspondiente solución. En un sistema DPS puro, todas las estrategias de interacción (cooperación y coordinación) se incorporan como una parte integral del sistema.

Para un Sistema Multiagente los diferentes sub-problemas se resuelven con agentes con sus propios intereses y objetivos, se podría decir que la investigación de éstos se orienta al estudio del comportamiento de un conjunto de agentes autónomos que tratan de dar solución a un problema dado.

La principal diferencia entre ambas áreas se encuentra en la flexibilidad de la coordinación entre los agentes. En la DPS, las interacciones y tareas que cada agente realiza están prefijadas de antemano: hay un plan centralizado de resolución del problema. Suele haber un miembro que ejerce un control global

que centraliza los resultados parciales y datos entre el resto de los componentes del sistema.

Por otro lado en los MAS, los agentes tienen un grado de autonomía mayor y pueden decidir dinámicamente qué interacciones son adecuadas, qué tareas deben realizar, quién realiza cada tarea y, además, es posible mantener conocimiento que no es globalmente consistente, incluso los agentes pueden mantener objetivos globales diferentes.

2.2.1. Definiciones

Los sistemas multiagente (SMA) según [JENNINGS, 1998] "*son sistemas de computadoras (hardware y/o software) compuestos por un conjunto de entidades (llamadas agentes) que cooperan a fin de satisfacer sus objetivos*".

Según [VAZQUEZ, 2004], "*…sistema formado por un conjunto de componentes (semi) autónomos que poseen las siguientes características:*

- *Cada agente no tiene información completa ni la capacidad para resolver el problema en su totalidad.*
- *Tienen puntos de vista limitados.*
- *No hay un sistema de control global.*
- *Los datos están descentralizados.*
- *Computación asíncrona."*

Entonces, de acuerdo a las definiciones expuestas, podemos decir que son sistemas compuestos por múltiples agentes que poseen un comportamiento propio y además se comunican (integran) con otros agentes para poder realizar tareas, entre ellos, de una mejor manera.

2.2.2. Arquitectura de un sistema multiagente

Los autores proponen diversas arquitecturas para los sistemas multiagente, pero una de estas arquitecturas, la propuesta por [VAZQUEZ, 2004], es la más completa y detallada, según mi criterio.

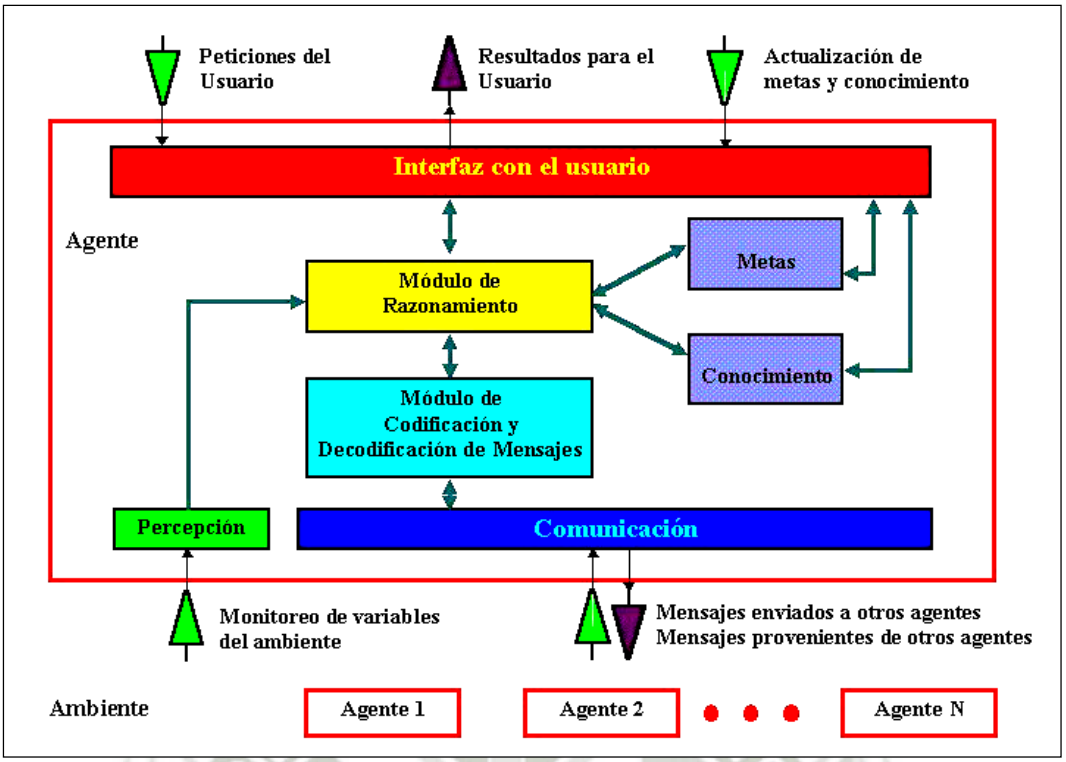


Figura 19: *Arquitectura genérica de un MAS.*

Fuente: *[VAZQUEZ, 2004].*

En la figura 03 podemos apreciar esta arquitectura genérica, la cual se detalla a continuación:

- *Interfaz con el usuario*: Se encarga de recibir requerimientos del usuario, enviarlos al módulo de razonamiento, y de presentar los resultados al usuario. Debe contar con los medios que permitan al usuario actualizar o modificar las metas y los conocimientos del agente.
- *Módulo de razonamiento*: Se encarga, en base al conocimiento y las metas del agente, de evaluar diferentes alternativas de solución además de negociar y seleccionar la mejor opción. Es capaz de tomar en cuenta los mensajes provenientes de otros agentes, o bien, de negociar con otros agentes. Puede, en base a los mensajes que reciba de otros agentes, o de la percepción que tenga de variables del ambiente, actualizar su base de conocimientos.
- *Metas*: Corresponden a los estados meta en la búsqueda de soluciones del agente.
- *Base de Conocimientos*: Se refiere a la información que tenga disponible el agente sobre la realidad que le rodea.

- *Módulo de codificación y decodificación de mensajes*: Este módulo se encarga de codificar los mensajes del agente en el formato de algún lenguaje de comunicación de agentes. También es capaz de recibir mensajes de otros agentes, decodificarlos (interpretarlos) y de enviar dichos mensajes al módulo de razonamiento. Como ejemplo de lenguaje de comunicación de agentes, puede citarse a KQML (**Knowledge Query and Manipulation Language**).
- *Módulo de percepción*: Se refiere a los medios con los que cuente el agente, para monitorear variables del medio ambiente que le rodea.
- *Módulo de comunicación*: Se encarga de enviar y recibir mensajes de otros agentes mediante protocolos de transporte (p. ej. en Internet, vía tcp y http).

2.2.3. Clasificación de los sistemas multiagente

La siguiente taxonomía, planteada por [STONE, 1997], está basada en dos aspectos de agentes considerados los más importantes: grado de heterogeneidad y grado de comunicación. La combinación de estos dos aspectos da origen a cuatro posibles escenarios:

i. *Sistemas Multiagente Homogéneos no Comunicativos*

Es el escenario multiagente más simple, este escenario está conformado por varios agentes con estructura idéntica (objetivos, posibles acciones y conocimiento del dominio), además tienen el mismo procedimiento para mapear sus acciones. Una diferencia son los sensores de entrada y los efectores; por lo cual están situados de forma diferente en el entorno y toman sus propias decisiones.

Los agentes de este escenario, no pueden comunicarse de manera directa entre ellos. En la figura 04 podemos apreciar esta primera clasificación.

ii. *Sistemas Multiagente Heteregóneos no Comunicativos*

Conformado por agentes heterogéneos, denominados así porque pueden tener distintos objetivos, modelos de dominio y/o acciones, y que no pueden comunicarse. Representado en la figura 05 con diferente fuente, tamaño y viñeta.

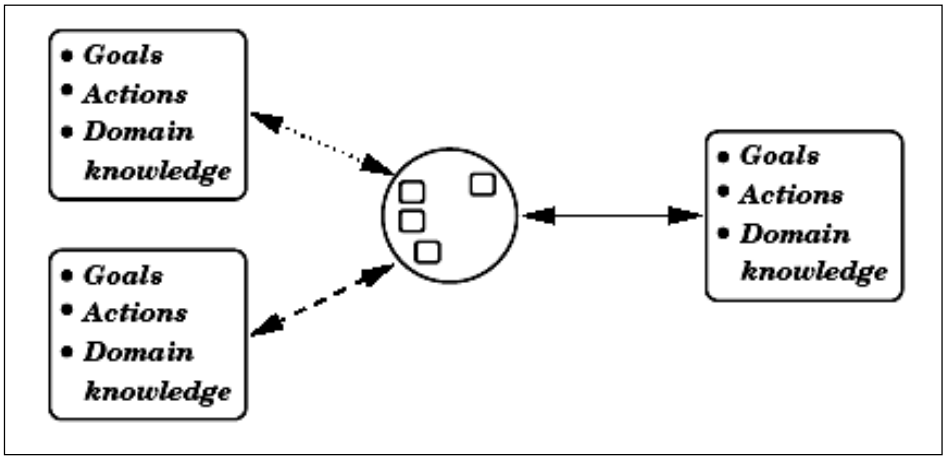


Figura 20: *MAS con agentes homogéneos no comunicativos.*

Fuente: *[STONE, 1997].*

Algo a destacar es el hecho de que los agentes heterogéneos pueden ser benevolentes (ayudar a los otros agentes) o competitivos (dedicarse solo a su función). Es decir, aun en el caso de que tengan diferentes objetivos pueden ser amigables con el resto o por el contrario intentar inhibirles.

Como en el caso del escenario homogéneo, los agentes están situados de manera diferente en el entorno, lo que hace que tengan entradas sensoriales diferentes y necesariamente tomaran acciones distintas.

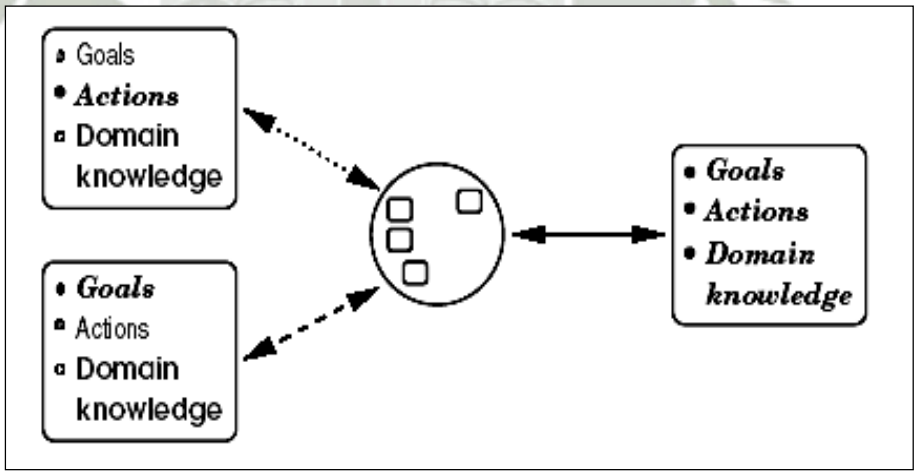


Figura 21: *MAS con agentes heterogéneos no comunicativos.*

Fuente: *[STONE, 1997].*

iii. Sistemas Multiagente Homogéneos Comunicativos

Son sistemas multiagente conformados por agentes idénticos que pueden comunicarse directamente con el resto, lo que permitirá una coordinación más efectiva.

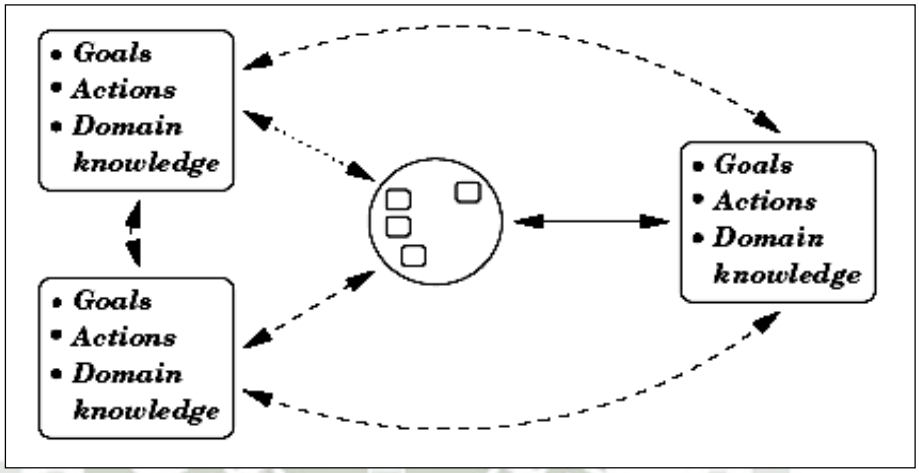


Figura 22: *MAS con agentes homogéneos comunicativos.*

Fuente: *[STONE, 1997].*

Algo importante de este escenario es que los agentes pueden comunicarse directamente, como indican las flechas en la figura 06.

La comunicación podría ponerse en una "pizarra" (un medio común) para ser interpretada por el resto de agentes, o bien podría ser dirigida de un agente a otro.

iv. Sistemas Multiagentes Heterogéneos Comunicativos

En el caso del escenario de agentes heterogéneos no comunicativos, estos podrían diferir en varias cosas: en los sensores de datos, en los objetivos, en las acciones y en el conocimiento del dominio. Por lo tanto los sistemas heterogéneos multiagente pueden ser muy complejos y poderosos. Sin embargo, el mayor poder de los SMA aparece cuando se añade la habilidad a los agentes heterogéneos, de comunicarse unos con otros.

En realidad, la habilidad de combinar comunicación y heterogeneidad implica la posibilidad de tener sistemas multiagente operando de forma similar a un sistema monoagente en términos de control.

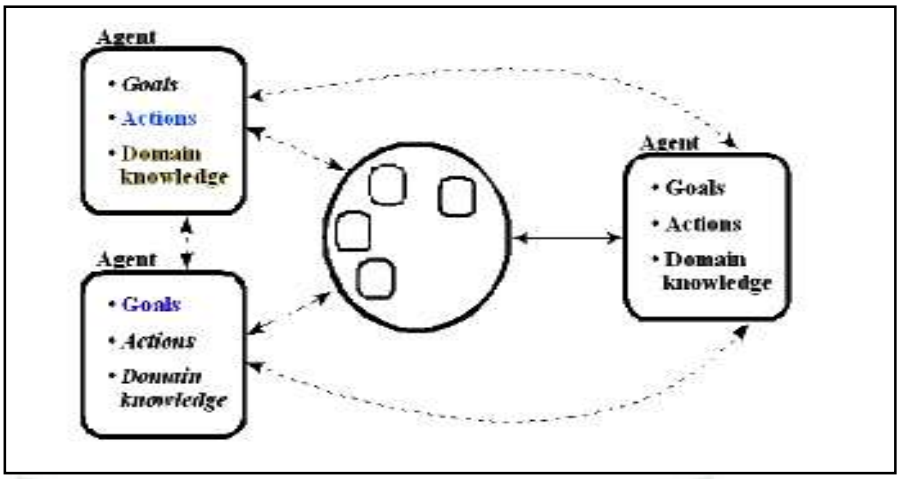


Figura 23: *MAS con agentes heterogéneos comunicativos.*

Fuente: *[STONE, 1997].*

2.2.4. Metodologías para sistemas multiagente

Existe diversidad de metodologías de desarrollo de software, las últimas y las más usadas son las metodologías orientadas a objetos. Pero hay que resaltar que existe una diferencia entre agente y objeto, debido a ello las metodologías orientadas a objetos no se adecuan a los sistemas multiagente.

Entre las metodologías más conocidas e "internacionales" tenemos: MAS-CommonKADS, Modelado y diseño de sistemas multiagente en BDI, GAIA, AUML, MESSAGE y MaSE. Los modelos de comunicación a analizar están basados en algunas de estas metodologías.

i. MAS-CommonKADS [IGLESIAS, 1998]

Esta metodología es una ampliación de la metodología CommonKADS y una combinación de técnicas de ingeniería del conocimiento, ingeniería del software orientada a objetos e ingeniería del software de protocolos. Esta metodología trabaja con siete modelos:

- Modelo de Agente
- Modelo de Tarea
- Modelo de la Experiencia
- Modelo de Coordinación
- Modelo de Comunicación
- Modelo de Organización
- Modelo de Diseño

Una ventaja importante de esta metodología es la interacción entre agentes, algo que hasta el tiempo de su creación no se daba.

ii. *GAIA [WOOLDRIDGE, 2000]*

Es una metodología de análisis y diseño de sistemas de agentes, una de sus principales ventajas es que es bastante general y comprensiva, además se puede aplicar a una gran cantidad de sistemas multiagente.

Toca el tema de los niveles, macro (sociedad) y micro (agente). Esta metodología considera a los sistemas multiagente como organismos computacionales que consiste en una interacción de varios roles.

iii. *MESSAGE [EURESCOM, 2000]*

Es una metodología que cubre las etapas de análisis y diseño de sistemas basados en agentes, además de explicar la relación que existe con la implementación y las pruebas del sistema.

Lo que hace esta metodología es extender las metodologías clásicas de desarrollo de software orientado a objetos para que puedan soportar el desarrollo de sistemas basados en agentes. Ofrece un método, un lenguaje de modelado, guías de aplicabilidad y recomendaciones.

iv. *MaSE [WOOD, 2000]*

Ingeniería del Software Multiagente, es una metodología que toma una especificación inicial del sistema y produce un conjunto de documentos formales en formato gráfico. Su principal objetivo es poder guiar al desarrollador por el ciclo de vida hasta conseguir el sistema basado en agentes.

Una ventaja importante de esta metodología es su independencia de la arquitectura y del lenguaje de implementación. Un sistema que se diseña con MaSE se puede implementar de diferentes maneras. Otra ventaja importante de esta metodología es su trazabilidad hacia delante y hacia atrás.

2.3 Modelos de comunicación[3]

No se puede hablar de la comunicación entre agentes sin preocuparnos de las propiedades de los agentes. Los agentes son comúnmente vistos en un nivel de descripción de intención. Por lo tanto los agentes residen en un nivel de conocimientos y por ende no pueden utilizar lenguajes generales y protocolos de computación distributiva adecuadamente. Este tipo de lenguajes y protocolos se centran en procesos más que en programas que constituyen a los agentes.

De lo anterior, un lenguaje de comunicación debe ser tan poderoso que sea capaz de soportar la comunicación entre programas de alto nivel y por otro lado los agentes tienen que enfrentar la tarea de traducir entre este nivel y el nivel de agente.

Se debe tener claro que un lenguaje de comunicación no es un protocolo, aunque ambos están relacionados a la comunicación. La diferencia entre lenguaje de comunicación y protocolo es bastante confuso. Un protocolo, desde el punto de vista de la comunicación, puede significar: (i) un protocolo de transporte, como http, SMTP, FTP, etc., (ii) un framework de alto nivel para la interacción, como Negociación, Protocolos de Teoría de Juegos, Planeamiento, etc., (iii) restricciones para las posibles sustituciones de las primitivas de comunicación. Un lenguaje de comunicación puede utilizar protocolos del primer tipo como mecanismo de transporte, puede utilizar protocolos del segundo tipo como medio de implementación, y usualmente incluye protocolos del tercer tipo como parte de su descripción, pero debe quedar claro que un lenguaje de comunicación no es solamente un protocolo.

Los lenguajes de comunicación deben cumplir algunos requerimientos básicos, entre ellos tenemos:

- *Forma*: Un buen lenguaje de comunicación debe ser declarativo, sintácticamente simple y legible por las personas.
- *Contenido*: Un buen lenguaje de comunicación se debe poder acomodar a otros sistemas.

[3] FININ, 1995. Páginas 7 – 9

- *Semántica*: Los desarrolladores deben tener un conocimiento compartido del lenguaje, de sus primitivas y de sus protocolos.
- *Implementación*: Su implementación debe ser eficiente en velocidad y ancho de banda.
- *Redes*: Se debe adaptar fácilmente a las tecnologías de redes.
- *Ambiente*: Tiene que adaptarse a los ambientes altamente distribuidos, heterogéneos y dinámicos de los agentes inteligentes.
- *Confiabilidad*: Debe proveer una comunicación confiable y segura.

2.3.1. Interacción entre agentes[4]

La interacción entre agentes se da por intercambio de información entre éstos. Este intercambio de información puede realizarse de 2 formas:

- Por la comunicación de mensajes que se establezca entre ellos.
- La percepción de cambios en el medio ambiente debidos a la acción de otro(s) agente(s).

De acuerdo al tipo de información que se intercambia podemos clasificar 3 tipos de interacción entre los agentes de un sistema multiagente:

i. Intercambio de conocimiento (interacción Baja)

En este tipo de interacción los agentes usan un conocimiento acerca de las percepciones que hayan podido obtener de su ambiente. Esta puede resultar en una percepción parcial de la realidad generándose 2 conocimientos, uno Complementario cuando es útil que cada agente pueda el razonar acerca del conocimiento de otros agentes para poder hacer el intercambio de información o Conflictivo en el cual se da un proceso de negociación para poder decidir.

ii. Intercambio de posibles soluciones (interacción media)

El intercambio de posibles soluciones sucede cuando dos o más agentes tienen que ponerse de acuerdo en una solución común o en un plan. El proceso consiste en encontrar la intersección es decir las decisiones en común que cada agente haya obtenido por separado, esta intersección podría ser vacía.

[4] Vázquez, 2004, Knowledge in Multiagent Systems: http://www.ijcai-03.org/~avazquez/knowledge/

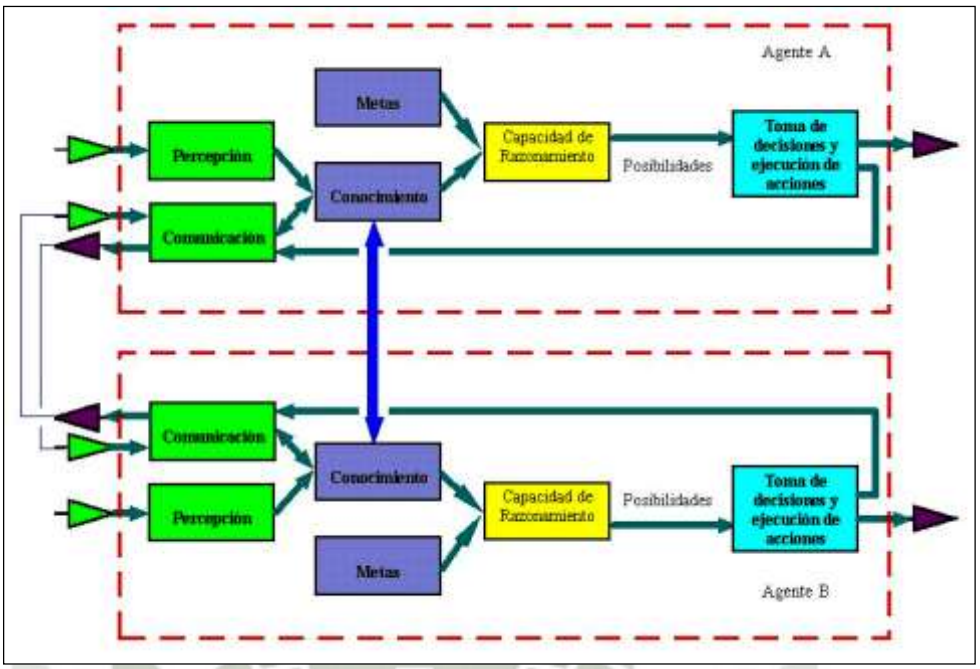


Figura 24: *Interacción baja.*
Fuente: *[VAZQUEZ, 2004].*

En este caso es útil que cada agente pueda razonar acerca de las capacidades de razonamiento de los otros agentes para deducir cual conocimiento debe comunicarles para producir posibles soluciones.

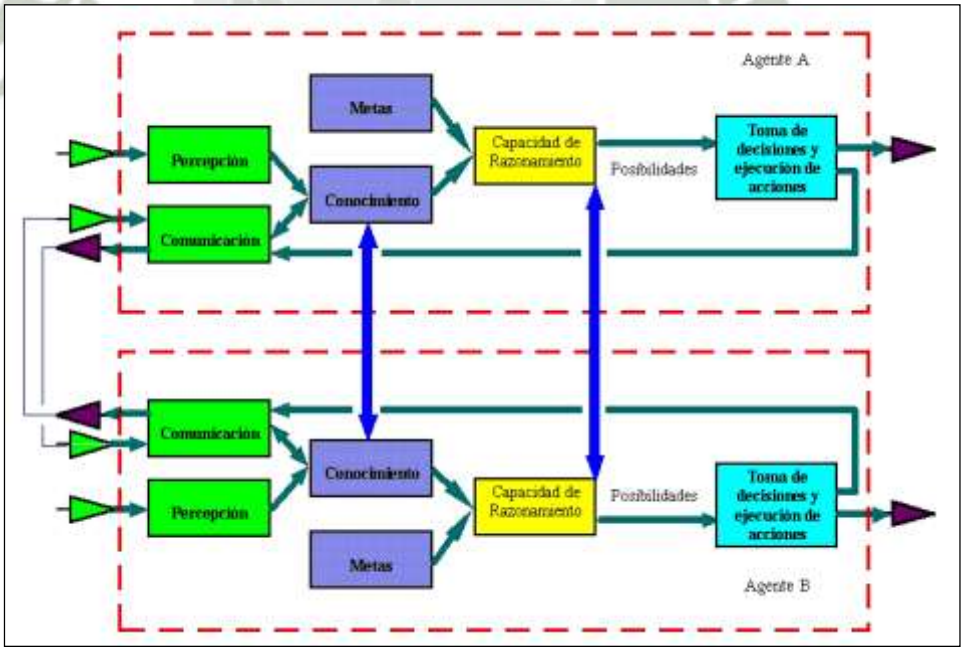


Figura 25: *Interacción media.*
Fuente: *[VAZQUEZ, 2004].*

iii. Intercambio de decisiones (interacción alta)

Cuando dos agentes logran encontrar decisiones posibles a un problema, existe un paso final que es el que los agentes se pongan de acuerdo entre las posibles soluciones al problema, puesto que una solución atractiva para uno de ellos puede que no sea la mejor para otro, por lo cual se entra a un proceso de negociación entre los agentes.

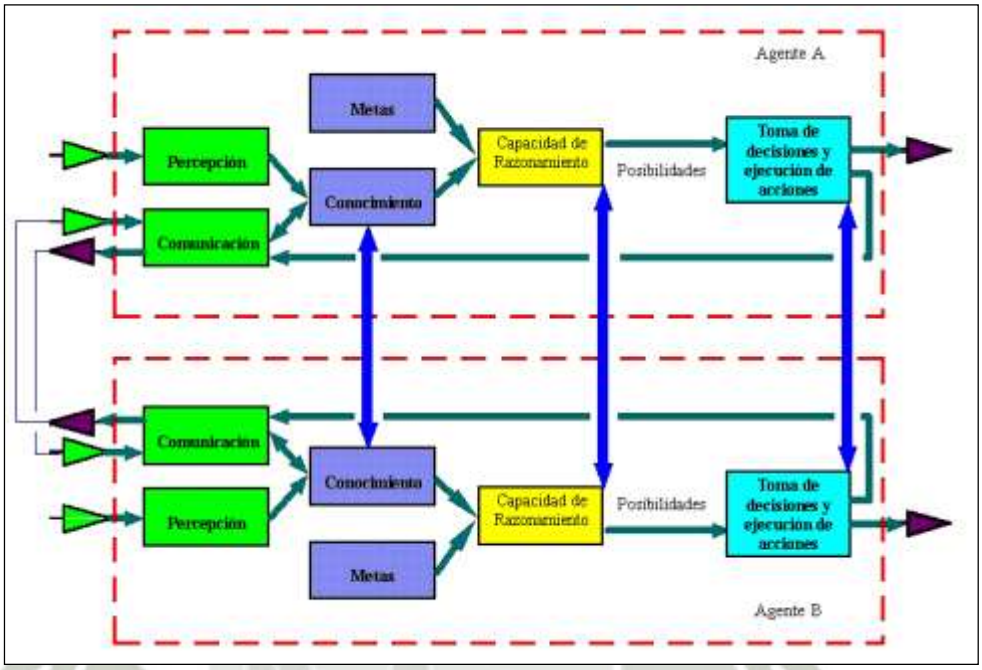


Figura 26: *Interacción alta.*
Fuente: *[VAZQUEZ, 2004].*

Algunas cosas importantes para resaltar en la interacción de agentes en un sistema multiagente, son: Cooperación, Coordinación, Negociación y Control. Detallaremos estos puntos en las siguientes secciones, de acuerdo a lo que plantea [STONE 1997].

- *Cooperación*

  En un sistema multiagente existen dos tipos de tareas que deben ser realizadas:

  - Tareas locales: Son las tareas relacionadas con los intereses individuales de cada agente.
  - Tareas globales. Son las tareas relacionadas con los intereses globales del sistema. Estas tareas globales son descompuestas y cada sub-tarea es realizada por un agente, de acuerdo a sus

habilidades y bajo el supuesto de que la integración de la solución de las sub-tareas, llevará a la solución global.

Existen varios modelos de cooperación, dentro de los cuales se pueden mencionar:

- Cooperación compartiendo tareas y resultados: Los agentes tienen en cuenta las tareas y los resultados intermedios de los demás para realizar las tareas propias.
- Cooperación por delegación: Un agente supervisor o maestro descompone una tarea en sub-tareas y las distribuye entre los agentes esclavos, para que sean resueltas. Después, el supervisor integra las soluciones para hallar la solución al problema inicial.
- Cooperación por ofrecimiento: Un agente maestro descompone una tarea en sub-tareas y las difunde en una lista a la que tienen acceso los agentes que integran el sistema, esperando que ellos ofrezcan su colaboración de acuerdo a sus habilidades. El supervisor escoge entre los ofrecimientos y distribuye las sub-tareas.

- *Coordinación*

  La coordinación entre un grupo de agentes les permite considerar todas las tareas a realizar y coordinarlas para no ejecutar acciones no deseables, por ejemplo:

  - Los agentes no generen y comuniquen sub-soluciones que lleven al progreso en la solución de un problema.
  - Los agentes generen y comuniquen resultados redundantes.
  - Distribución inapropiada de la carga de trabajo entre los agentes.

  Hay varios modelos de coordinación de acciones entre agentes, pero dos principales (que se mencionan a continuación) y gran cantidad de modelos intermedios.

  - *Coordinación Global*: Cuando el sistema multiagente determina y planifica globalmente las acciones de los diferentes agentes.
  - *Coordinación Individual*: Cuando el sistema multiagente le da completa autonomía a los agentes, es decir, cada agente decide qué hacer y resuelve localmente los conflictos que detecte con otros agentes.

- *Negociación*

  Para que los mecanismos de cooperación y coordinación sean exitosos en un sistema de agentes que actúan interdependientemente, debe existir un mecanismo adicional, por medio del cual, los integrantes de un sistema se puedan poner de acuerdo cuando cada agente defiende sus propios intereses, llevándolos a una situación que los beneficie a todos teniendo en cuenta el punto de vista de cada uno. Este mecanismo es llamado negociación.

  Los procesos de negociación tienen como resultado la modificación o confirmación de las creencias de cada agente involucrado, en lo relacionado con los demás agentes y con el mundo en el que se desenvuelve.

  Para que una negociación sea exitosa es necesario un protocolo que facilite y en lo posible garantice la convergencia de ideas a una solución común. Un protocolo establece un conjunto de pasos que debe seguir un proceso de negociación, así como las posibles respuestas de un agente, a las acciones de otro agente.

- *Control*

  El control es el mecanismo básico que provee apoyo para la implementación de mecanismos de coordinación en un sistema multiagente. El control se relaciona directamente con:

  - Determinar cuáles son las sub-tareas más importantes a realizar en un momento dado.
  - Determinar qué contexto (resultados intermedios de otros agentes) deben ser usados en la solución de una sub-tarea.
  - Estimar el tiempo de generación de la solución a una sub-tarea.
  - Evaluar si la solución de un problema ha sido generada. (Problema de la terminación).

El control puede ser considerado desde dos puntos de vista: control global y control local. El control global se relaciona con tomar decisiones basándose en datos obtenidos y consolidados a partir de la información de todos los agentes del sistema, el control local se relaciona con tomar decisiones basándose solo en datos locales.

2.3.2. Organización Social[5]

Es la manera como el grupo de agentes está constituido en un instante dado. La organización social está relacionada con la estructura de los componentes funcionales del sistema, sus características, sus responsabilidades, sus necesidades y la manera como realizan sus comunicaciones. Esta organización puede ser estática o dinámica, dependiendo de las funciones o tareas de cada agente. Se puede considerar que una *sociedad de agentes* está constituida por tres elementos:

- Un grupo de agentes.
- Un conjunto de tareas a realizar.
- Un conjunto de recursos.

La organización en los sistemas multiagente depende del tipo de comunicación y el modo de cooperación entre agentes, así como del tipo de agentes que conforman el grupo. En general se pueden distinguir tres tipos de configuraciones organizacionales, y una combinación de las anteriores:

- *Estructura Centralizada*: En este tipo de configuración existe un agente que controla la interacción de los demás agentes del sistema, porque tiene la información o la funcionalidad para hacerlo.
- *Estructura Horizontal*: Este tipo de configuración existe cuando todos los agentes que integran un sistema están al mismo nivel, es decir, no hay ningún agente que haga las veces de maestro o supervisor, ni tampoco agentes esclavos.
- *Estructura Jerárquica*: Esta configuración existe cuando los agentes trabajan diferentes niveles de abstracción de un problema, es decir, la configuración es de niveles. En un mismo nivel se establece una configuración horizontal, si hay más de un agente. Para resolver un problema cada agente divide el

[5] Stone-Veloso, 1997, Multiagent Systems: A Survey from a Machine Learning Perspective: http://www-2.cs.cmu.edu/afs/cs/usr/pstone/public/papers/97MAS-survey/revised-survey.html

problema en sub-problemas que él puede resolver, sub-problemas que puede resolver con la cooperación de los agentes que están al mismo nivel y sub-problemas que sabe que los agentes de niveles inferiores de la jerarquía pueden resolver.

- *Estructura "ad hoc"*: Esta configuración puede ser una mezcla de las tres anteriores, se caracteriza porque lo dinámico de la estructura está regido por el ajuste mutuo entre los pequeños grupos de agentes en el sistema.

2.3.3. Requerimientos para la comunicación entre agentes[6]

Para poder establecer la comunicación entre un agente y otros agentes podemos definir tres componentes básicos: los componentes de representación, los componentes de comunicación y los componentes no relacionados con el conocimiento compartido. La relación entre los mismos la podemos ver en la figura 11.

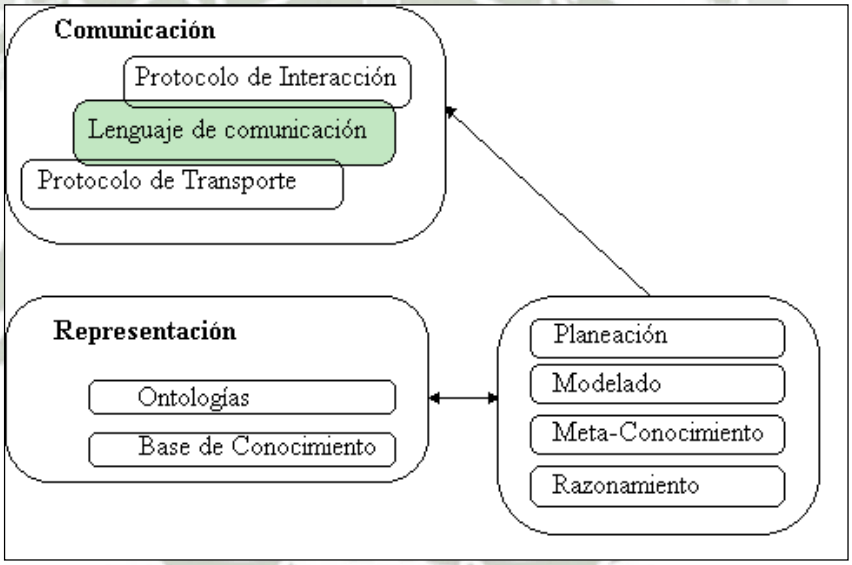


Figura 27: *Componentes básicos de la comunicación.*

Fuente: *[VAZQUEZ, 2004].*

i. Componentes de representación

Incluyen las ontologías (Vocabularios estructurados que comparten los interlocutores) y las Bases de Conocimiento. Este tipo de componentes son los que hacen posible resolver el problema del entendimiento mutuo.

ii. Componentes de comunicación

Los componentes de comunicación constan de tres elementos importantes:

[6] Vázquez, 2004, Knowledge in Multiagent Systems: http://www.ijcai-03.org/~avazquez/knowledge/

- *Un protocolo de transporte*: Mecanismo de transporte usado para la comunicación.
- *Un lenguaje común*: Medio por el cual las actitudes acerca del contenido del intercambio se comunican, es decir, poder saber si el contenido de la comunicación es una pregunta, una aseveración o una solicitud.
- *Un protocolo de interacción*: Es la estrategia del agente para interactuar con otros agentes.

iii. *Componentes no relacionados con el entendimiento compartido*

Son componentes que no participan directamente en un proceso de comunicación, sin embargo ayudan al Agente a poder desempeñarse mejor en sus tareas como por ejemplo la habilidad de razonar sobre sus propias acciones, capacidad de representar una meta, planificar actividades y modelar otros agentes.

2.3.4. Formas de Comunicación

La forma en la que los agentes pueden comunicarse depende de la manera que estos puedan ser organizados, a continuación se describirán dos formas:

i. *Comunicación Directa*

En este tipo de organización los agentes manejan su propia organización, tiene la ventaja de que no depende de otros programas, sin embargo su complejidad de implementación se incrementa considerablemente. En este tipo de comunicación podemos definir dos tipos de arquitecturas:

- *Arquitectura de red de contratos*: Ante la necesidad de información los agentes distribuyen peticiones hacia otros agentes receptores estos receptores evalúan las solicitudes y mandan ofertas hacia el agente solicitante y al final este evalúa las ofertas presentadas por los agentes receptores y realiza un contrato. Es una arquitectura de alta coordinación pero es muy costosa por la cantidad de mensajes que deben procesarse para cerrar un ciclo de comunicación.

- *Arquitectura de especificación compartida*: En este tipo de arquitectura no se hacen solicitudes de servicio. Los agentes proveen información a otros de sus capacidades y necesidades y cuando surge la necesidad de un servicio esta información es utilizada por los agentes para coordinar sus actividades.

ii. Coordinación Asistida

La comunicación de un agente a otro se hace por intermedio de facilitadores, este facilitador es el que se encarga de encontrar la ruta para hacer llegar la solicitud a otros facilitadores y a algún agente de dominio que pueda satisfacer la solicitud.

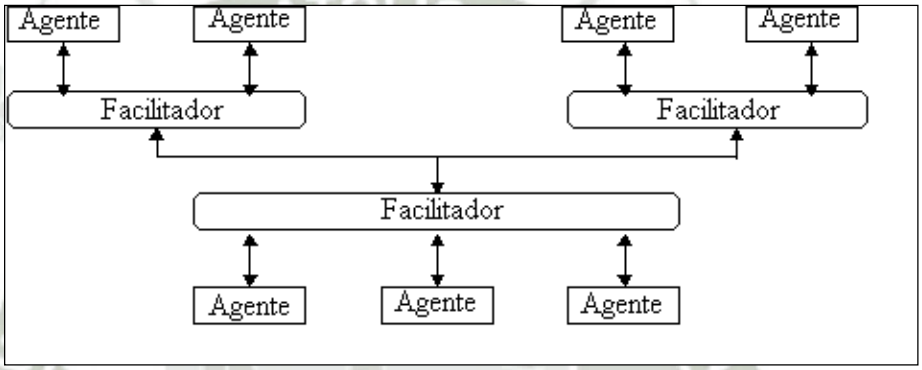


Figura 28: *Comunicación con coordinación asistida.*

Fuente: *[VAZQUEZ, 2004].*

2.3.5. Comunicación en sistemas multiagente

En todo sistema multiagente [IGLESIAS, 1998], sus componentes principales (los agentes) pueden mejorar su coordinación y coherencia gestionando qué, cómo y cuándo se comunican entre sí. La comunicación puede proporcionar a los agentes el conocimiento necesario para desarrollar sus acciones con una visión menos local y poder sincronizarse con el resto de agentes. Sin embargo, una excesiva comunicación puede dar lugar a una sociedad de agentes burocráticos, cuya sobrecarga de comunicación sea mayor que el trabajo efectivo realizado. Se puede distinguir un rango amplio de formas de comunicación [WERNER, 1989], que van desde la falta de comunicación hasta la comunicación de alto nivel y la interacción hombre máquina.

En los siguientes párrafos se procederá a revisar estas formas de comunicación, según lo establecido en [WERNER, 1989].

i. Sin comunicación

Los agentes pueden interactuar sin comunicarse, infiriendo las intenciones de otros agentes [ROSENSCHEIN, 1988]. Esta situación puede darse debido a fallos de hardware, a la imposibilidad de comunicarse o al deseo de que los agentes gocen de una mayor autonomía. Normalmente, para que sea posible la cooperación se supondrá que los agentes disponen de la información sensorial suficiente para poder inferir los objetivos e intenciones del resto de agentes. Para estudiar este tipo de interacción científicos han recurrido a la teoría de juegos, empleado matrices de costes, en las que se representa la ganancia que obtiene cada agente dependiendo de la acción que realice otro agente. Este enfoque se ha extendido con técnicas probabilísticas para representar la incertidumbre sobre los posibles movimientos de los otros agentes.

ii. Comunicación primitiva

La comunicación se restringe a un número de señales con interpretaciones fijas (p.e. el uso de las señales enviar/recibir en un proceso secuencial de comunicación). Este trabajo ha sido aplicado a planificación multiagente [GEORGEFF, 1989] para coordinar a dos agentes que tienen un conflicto (de escasez de recursos) mediante un mediador, pero la coordinación que puede lograrse es limitada (ceder el turno a través de un mediador que sincroniza a los agentes).

iii. Arquitectura de pizarra

La arquitectura de pizarra [CARVER, 1992] [CORKILL, 1991] es otro paradigma de comunicación, cuya estructura consta de tres componentes principales:

a) La pizarra.
b) Un conjunto de fuentes de conocimiento (KSs, Knowledge Sources).
c) Un mecanismo de control.

La pizarra es una base de datos global (compartida por todas las KSs) que contiene datos e hipótesis (soluciones parciales potenciales). Se suele estructurar la pizarra en niveles, asociando clases de hipótesis a un nivel, y las hipótesis suelen estar ligadas a hipótesis de otro nivel. Las fuentes de conocimiento son módulos especializados que intercambian mensajes (hipótesis, resultados parciales, etc.) a

través de la pizarra. Partiendo de los resultados e hipótesis de otras KSs, una KS puede generar otra solución parcial tentativa. Cada nivel de la pizarra suele suponer un nivel de abstracción en la resolución del problema, y las KSs trabajando en un nivel ven las hipótesis de su nivel y los adyacentes, de forma que las hipótesis de los niveles próximos se van propagando a los niveles más abstractos de forma ascendente.

Para gestionar la concurrencia en el acceso a la pizarra se debe emplear un mecanismo de control, típicamente basado en una agenda. Esta agenda es gestionada por un monitor de la pizarra que activa las fuentes de conocimiento adecuadas según los datos mantenidos en la pizarra. Algunas arquitecturas de agente utilizan esta arquitectura para intercambiar información entre subsistemas de agentes o para modelar el intercambio entre los distintos módulos (o subagentes) que componen la estructura de un agente.

iv. Paso de mensajes

Los sistemas de paso de mensajes permiten que un agente envíe un mensaje (tanto de peticiones de servicios e información como de respuesta a dichas peticiones) a uno o más agentes cuyos nombres debe conocer. A diferencia de la arquitectura de pizarra, los agentes deben mantener conocimiento sobre su entorno para saber a qué agentes deben dirigir sus mensajes. Este modelo tiene su origen en la programación orientada a objetos concurrente.

Numerosos sistemas multiagente han adoptado el paso de mensajes definiendo un formato para dichos mensajes, tipos de mensaje y un protocolo para procesar dichos mensajes.

v. Comunicación de alto nivel

Existen diversos trabajos de investigación para estudiar las interacciones entre los agentes en el nivel de conocimiento [NEWELL, 1982] en vez de en el nivel simbólico. Esto supone que los agentes puedan razonar sobre las intenciones, deseos y objetivos de otros agentes, y que puedan comunicar estos deseos, objetivos e intenciones.

Para analizar estas interacciones complejas, se han intentado aplicar a los sistemas multiagente algunas técnicas y teorías provenientes del campo del lenguaje natural, en especial de análisis y generación del discurso. De acuerdo con

las técnicas simbólicas de comprensión del diálogo, no basta con analizar el significado de cada frase para entender un texto, sino que es necesario comprender las intenciones de los interlocutores y cómo se desarrolla el diálogo para satisfacer sus objetivos, ya que no toda la comunicación suele ser explícita.

vi. Interacción hombre-máquina

La comunicación entre un agente artificial y un agente humano ha tenido en los últimos tiempos gran relevancia. Básicamente, se han tomado dos aproximaciones: encapsular al agente humano modelando sus interacciones en un lenguaje de comunicación de agentes o aprovechar la tecnología multiagente para simplificar las interfaces hombre-máquina.

En el modelo de comunicación propuesto por CommonKADS se permitía cualquier tipo de interacción de un agente con el exterior (otros agentes, sensores, sistema operativo, etc.). En MAS-CommonKADS [IGLESIAS, 1998] el modelo de comunicación, sólo se encarga del modelado de las relaciones hombre-máquina. Por lo tanto:

- Las interacciones con otros agentes no humanos son modeladas en el modelo de coordinación.
- Las señales recibidas del exterior son modeladas mediante una entidad genérica denominada entorno. En el modelo de organización se muestran todas las relaciones de cada agente con el exterior, esto es, el entorno y el resto de agentes. El modelado de los sensores para procesar las señales del entorno se realiza en el modelo de agente. Si la recepción de una señal inicia una conversación con otros agentes, esta conversación es modelada en el modelo de coordinación.
- Las denominadas conversaciones reactivas, consistentes en que los agentes se comunican indirectamente a través de las modificaciones que realizan en el entorno, no son abordadas en este modelo si no conllevan una comunicación directa.

2.3.6. Knowledge Query Manipulation Language (KQML)[7]

Este es uno de los lenguajes más utilizados para lacomunicación entre agentes. Veamos una breve explicación, del autor, sobre este lenguaje. KQML fue concebido como un formato de mensaje y como un protocolo para manejar mensajes para que pueda apoyar el intercambio de conocimientos entre los agentes en tiempo real. Las características de KQML pueden resumirse en las siguientes:

- Los mensajes KQML son opacos al contenido que transportan. Los mensajes KQML no se limitan a comunicar sentencias en algún idioma, sino que más bien comunican una actitud acerca del contenido (afirmación, petición, consulta).
- Las primitivas del lenguaje se llaman Performativas. Como sugiere el término, el concepto está relacionado con la teoría del habla. Las Performativas definen las acciones permitidas (operaciones) que los agentes pueden tratar al comunicarse con los demás.
- Un entorno de agentes que hablen KQML puede ser enriquecido con agentes especiales, llamados facilitadores, que proporcionan funciones tales como: asociación de direcciones físicas con nombres simbólicos; registro de las bases de datos y/o servicios ofrecidos y buscados por los agentes, y servicios de comunicación (transmisión, intermediación, etc.) Utilizando una metáfora, los facilitadores actuar como secretarias eficientes de los agentes en su dominio.

La interacción inteligente es más que un intercambio de mensajes. El KQML es un intento para desligar estos temas de los lenguajes de comunicación, que deben definir un conjunto estándar de tipos de mensajes que se han de interpretar de forma idéntica por todos los entes que interactúan. Un lenguaje universal de comunicación es de mucho interés para una amplia gama de aplicaciones que necesitan comunicar algo más que oraciones pre-definidas o fijas de los hechos. KQML es la pieza central de esta presentación.

Para hacer frente a muchas de las dificultades de la comunicación entre los agentes inteligentes, se les debe dar un lenguaje común. En términos lingüísticos,

[7] Finin, 1995. Secciones 3.3 y 5.

esto significa que deben compartir una sintaxis, semántica y pragmática. Obtener procesos de información o agentes de software que tengan una sintaxis en común es un problema de mucha relevancia. No existen lenguajes universalmente aceptados en los cuales poder representar información y consultas. Existen varios lenguajes con sus respectivos partidarios, pero también hay una fuerte posición de que es demasiado pronto para estandarizar la representación en cualquier lenguaje. Por el momento es necesario decir que dos agentes pueden comunicarse entre sí si tienen un lenguaje común de representación o que utilizan lenguajes que se pueden traducir en ambos lados de la comunicación. Asumiendo la utilización de un lenguaje común o traducible, sigue siendo necesario para la comunicación entre los agentes compartir un framework de conocimientos, a fin de interpretar los mensajes que intercambian. Esto realmente no es una semántica compartida , sino más bien una ontología compartida. Es poco probable que se comparta una ontología, sino muchas. Las ontologías compartidas se están desarrollando en muchos dominios importantes de aplicación, tales como la planificación y la programación, la biología y la medicina. La pragmática de la comunicación de agentes de software implica cuestiones como saber con quién hablar y cómo encontrarlos, así como saber cómo iniciar y mantener un intercambio. KQML se ocupa principalmente de este tipo de pragmática y secundariamente con la semántica. Es un lenguaje y un conjunto de protocolos de software que ayuda en la identificación de los agentes, con la conexión y el intercambio de información con otros de estos agentes.

El lenguaje KQML puede ser visto como una composición de tres capas: la capa de contenido, la capa de mensaje, y la capa de comunicación.

- La capa de contenido lleva el contenido real del mensaje, en el mismo lenguaje de representación de los programas. KQML puede llevar expresiones codificadas en cualquier lenguaje de representación, incluyendo lenguajes expresados como cadenas ASCII y aquellas expresadas utilizando una notación binaria. Algunos agentes que hablen KQML (por ejemplo, enrutadores, intermediarios muy generales, etc.) pueden pasar por alto la porción de contenido de los mensajes, salvo para determinar dónde termina.

- La capa de comunicación codifica un conjunto de características del mensaje que describen los parámetros de comunicación de bajo nivel, tales como la identidad del remitente y el receptor, y un identificador único asociado con la comunicación.
- La capa de mensaje es utilizada para codificar un mensaje que una aplicación desea transmitir a otra. La capa de mensaje constituye el núcleo del KQML, y determina el tipo de interacciones se puede tener con un agente que hable KQML. Una función importante de la capa de mensaje es identificar el protocolo que se utilizará para entregar el mensaje y para proveer una performativa con la cual el remitente adjunta los contenidos (tal como una afirmación, una pregunta, una orden, o cualquiera del conjunto de performativas conocidas). Además, dado que el contenido puede ser opaco para el agente, esta capa incluye características opcionales que describen el contenido del lenguaje, la ontología que asume, y algún tipo de descripción de su contenido, como un descriptor de nombres de un tema dentro de la Ontología. Estas características hacen posible que KQML pueda analizar, rutear y entregar correctamente los mensajes a pesar de que su contenido es inaccesible.

La sintaxis de KQML se basa en una lista balanceada de paréntesis. El primer elemento de la lista es la performativa, los demás elementos son los argumentos de la performativa como un par palabra clave/valor. Debido a que el lenguaje es relativamente simple, la sintaxis no es significativa y puede ser modificada en caso necesario en el futuro. La sintaxis revela las raíces de las primeras implementaciones, que se realizaron en Common Lisp, que ha resultado ser bastante flexible.

Un mensaje de KQML del agente Joe que representa una consulta sobre el precio de una cotización de IBM podría ser codificada como:

```
(ask-one
:sender joe
:content (PRICE IBM ?price)
:receiver stock-server
:reply-with ibm-stock
```

:language LPROLOG
:ontology NYSE-TICKS)

En este mensaje, la performativa KQML es *ask-one*, el contenido es *(price ibm ?price)*, la ontología asumida por la consulta es identificada por el token *nyse-ticks*, el receptor del mensaje ha de ser un servidor identificado como un *stock-server* y la consulta está escrita en un lenguaje llamado *LPROLOG*. El valor de la palabra clave *:content* es el nivel de contenido, los valores de *:reply-with, :sender* y *:receiver* forman la capa de comunicación y el nombre de la performativa, con el *:language* y *:ontology* forman la capa de mensaje.

2.4. Web Services

Según [MSDN, 2008], "*Un Servicio Web es un componente software que se comunica con otros componentes, a través de mensajes codificados en XML, utilizando protocolos de transporte estándares de Internet, como SMTP o HTTP, que son capaces de atravesar perfectamente los cortafuegos. Intuitivamente, un Servicio Web es ese sitio Web "caja negra" sin interfaz de usuario que puede ser reutilizado para dar servicio a programas en lugar de a personas*."

Otra definición, es la ofrecida por [WIKIPEDIA, 2008], "*Un servicio web (en inglés Web Service) es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.*"

La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web. Para mejorar la interoperabilidad entre distintas implementaciones de servicios Web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares."

2.4.1. Estándares empleados

El conjunto de servicios y protocolos de los servicios Web se denominan Web Services Protocol Stack (WSPS), entre los cuales tenemos:

- XML (Extensible Markup Language): Es el formato estándar para los datos que se vayan a intercambiar.
- SOAP (Simple Object Access Protocol) o XML-RPC (Extensible Markup Language – Remote Producer Call): Protocolos sobre los que se establece el intercambio.
- WSDL (Web Services Description Languages): Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.
- UDDI (Universal Description, Discovery and Integration): Protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios web están disponibles.
- WS-Security (Web Service Security): Protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

Entre otros protocolos de transferencia de datos también tenemos:

- HTTP (Hypertext Transfer Protocol)
- FTP (File Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)

2.4.2 ¿Por qué utilizar Servicios Web?

La principal razón para usar servicios Web es que se basan en HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante firewalls (que filtran y bloquean gran parte del tráfico de Internet), cierran casi todos los puertos TCP salvo el 80, que es, precisamente, el que usan los navegadores. Los servicios Web se enrutan por este puerto, por la simple razón de que no resultan bloqueados.

Otra razón es que, antes de que existiera SOAP, no había buenas interfaces para acceder a las funcionalidades de otros ordenadores en red. Las que había eran ad

hoc y poco conocidas, tales como EDI (Electronic Data Interchange), RPC, u otras Application Programming Interface APIs.

Una tercera razón por la que los servicios Web son muy prácticos es que pueden aportar gran independencia entre la aplicación que usa el servicio Web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más acusada.

2.4.3. Ventajas

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, los servicios Web pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar.

2.4.4. Inconvenientes

- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture).
- Su rendimiento es bajo si se compara con otros modelos de computación distribuida, tales como RMI (Remote Method Invocation), CORBA, o DCOM (Distributed Component Object Model). Es uno de los inconvenientes derivados de adoptar un formato basado en texto. Y es que entre los objetivos de XML no se encuentra la concisión ni la eficacia de procesamiento.
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.

- Existe poca información de servicios web para algunos lenguajes de programación.

3. ANTECEDENTES INVESTIGATIVOS

Se ha realizado una búsqueda en bibliotecas, hemerotecas, bases de datos científicas como: Engineering Village 2, Web of science, ACM Digital Library, IET (IEEE Electronic Library). Además en buscadores científicos como Scirus y Scholar Google y se han encontrado los siguientes artículos relacionados al tema:

Autonomic communication services: a new challenge for software agents (2008)

Autores: Raffaele Quitadamo, Franco Zambonelli.

Propósito: Analizar la definición de modelos para servicios de comunicación autonómicos y discutir la relación de estos modelos y los modelos de agentes.

Resultados: Presentan los resultados de realizar una encuesta de los diferentes modelos para comunicación autónoma. Una de las conclusiones más importantes del artículo es que los sistemas de agentes y sistemas multiagentes deberían son los que deben liderar la búsqueda de un modelo de comunicación autónomo, por la naturaleza misma de este tipo de sistemas.

Por lo tanto podemos agregar a esa conclusión que los modelos de comunicación deben evolucionar y buscar un mecanismo basado en servicios que haga que esta comunicación sea más autónoma.

A multiagent system for assisting citizens in their search of e-government services (2008)

Autores: Pasquale De Meo, Giovanni Quattrone, Domenico Ursino.

Propósito: Mostrar un sistema multiagente de apoyo a ciudadanos en un ambiente de gobierno electrónico (basado en web).

Resultados: Se demuestra la eficiencia en el servicio ofrecido, para ello se pone bastante énfasis en la coordinación que deben tener los agentes participantes. Por lo tanto el modelo de comunicación debe ser efectivo para lograr el objetivo buscado.

Internet-based integrated environmental assessment, part II: Semantic searching based on ontologies and agent systems for knowledge discovery (2006)

Autores: Steven Kraines, Rafael Batres, Brian Kemper, Michihisa Koyama, Vincent Wolowski.

Propósito: Mostrar un prototipo web basado en multiagentes para compartir conocimiento que permita solucionar problemas en sistemas de gran escala.

Resultados: El sistema propuesto simula un ecosistema urbano que usa la combinación de una ontología de alto nivel con un dominio específico. Usa un modelo de comunicación que pretende emular avisos comerciales que es manipulado por los agentes.

Algo importante a resaltar es que no se hace uso de web services, pero si del concepto de servicios.

A spatially dependent communication model for ubiquitous systems (2005)

Autores: S. Bandini, S. Manzoni, G. Vizzari.

Propósito: Proponer un modelo de sistema multiagente multicapa como arquitectura conceptual de sistemas ubicuos.

Resultados: El artículo concluye que los modelos y conceptos necesarios para la computación ubicua tienen como elemento fundamental la interacción entre los agentes (comunicación). Señalan que dos elementos fundamentales para esta interacción son: (1) la reacción, y (2) la emisión de un campo que representa la comunicación en sí. Se recalca que los sistemas multiagentes e Internet son de vital importancia para este tipo de computación, así como de la tecnología de los web services.

Web-enabling MultiAgent systems (2004)

Autores: E. Ramirez, R. Brena.

Propósito: Proponer una arquitectura para la interacción de los sistemas web con los usuarios finales (p.e. a través de un portal web).

Resultados: Lo importante de este artículo radica en que la capacidad que se da a los agentes para publicar sus resultados a través de web services o en HTML plano, aprovechando la facilidad que ofrecen los lenguajes actuales para implementar este tipo de interfaces de salida.

Se debe recalcar que la arquitectura sirve para la salida del sistema mas no para la interacción entre los agentes.

A framework of multi-agent-based modeling, simulation, and computational assistance in an ubiquitous environment (2004)

Autores: K. Shibuya.

Propósito: Proponer un framework que explore el comportamiento humano y que permita desarrollar servicios en un ambiente de computación ubicua.

Resultados: Lo importante de este artículo no radica en el framework que se ofrece, sino que contribuye al desarrollo de diversos servicios para computación ubicua. Estos servicios integran sistemas multiagente y los web services, con esta integración se busca modelar, simular y dar asistencia a sistemas de computación ubicua.

Se debe recalcar que lo importante para el presente trabajo de investigación es la existencia de trabajos que utilizan web services para desarrollar sistemas multiagentes, pero no los utilizan para la interacción de los agentes, sino para las tres actividades señaladas (modelado, simulación y asistencia).

Implementation of an ontology sharing mechanism for multiagent systems based on web services (2004)

Autores: L. Sheremetov, M. Contreras, A. Smirnov.

Propósito: Proponer una integración de tecnología de agentes con la funcionalidad que ofrecen los web services para el intercambio de conocimiento.

Resultados: Es una de las primeras propuestas de integrar la tecnología de agentes con la funcionalidad de los web services. El intercambio de conocimiento se basa en una ontología de propósito general. Por lo tanto el centro de este trabajo de investigación se centra en el desarrollo de la ontología más que en la utilización de los web services como mecanismo de interacción. Se explica el mecanismo de implementación de este tipo de integración pero no se profundiza en modelos específicos, dejando a la capacidad del desarrollador el implementar los diferentes modelos con las herramientas que más utilicen.

A perspective on multiagent coordination models (2003)

Autores: L. Bocchi L, P. Ciancarini.

Propósito: Realizar una revisión del estado del arte de los modelos de comunicación y lenguajes de sistemas multiagentes.

Resultados: Una de las conclusiones señala que es importante adaptar diferentes modelos de coordinación a infraestructuras de redes, las cuales son llamadas arquitecturas pre-WWW y post-WWW. Otra conclusión remarca que los web services serán el punto de partida para nuevos modelos de coordinación.

4. OBJETIVOS

4.1. Principal

Proponer un Modelo de comunicación para sistemas multiagente utilizando la tecnología Web Services.

4.2. Secundarios

- Determinar la propuesta del modelo de comunicación.
- Determinar el mejoramiento del modelo mediante la utilización de los web services.

5. HIPÓTESIS

Debido a la complejidad de entendimiento e implementación de los modelos de comunicación para Sistemas Multiagente existentes, se puede plantear un nuevo modelo de comunicación utilizando Web Services, de tal forma que supere los problemas de claridad, implementación y adaptabilidad que poseen algunos de los modelos propuestos hasta el momento.

III. PLANTEAMIENTO OPERACIONAL

3.1. TÉCNICAS, INSTRUMENTOS Y MATERIALES DE VERIFICACIÓN

La técnica a utilizar es la Encuesta y los instrumentos serán el Cuestionario estructurado, indirecto y colectivo y un simulador que nos permita ver el funcionamiento del modelo propuesto.

En la siguiente página se muestra el cuestionario a utilizar para levantar los datos.

MODELO DE COMUNICACIÓN PARA SISTEMAS MULTIAGENTE UTILIZANDO WEB SERVICES

Después de leer y analizar el modelo planteado responda las siguientes preguntas.

1. ¿Cómo considera Ud. el nivel de entendimiento del modelo de comunicación?

   Muy Claro ___ Claro ___ No Claro ___

2. Después de utilizar el simulador, ¿le quedó claro el modelo?

   Muy Claro ___ Claro ___ No Claro ___

3. Desde su perspectiva, ¿es sencillo el modelo planteado?

   SI ___ NO ___

4. Basándose en su experiencia, ¿cómo considera Ud. la implementación del modelo de comunicación?

   Muy Fácil ___ Fácil ___ Regular ___ Difícil ___ Muy Difícil __

5. ¿Considera Ud. que el modelo tiene todo lo necesario para complementar la implementación de un sistema multiagente?

   SI ___ NO ___

6. ¿Cree Ud. que el modelo se adapte y sea entendido por todo tipo de usuario?

   SI ___ NO ___

7. Si su respuesta anterior fue SI, responda la pregunta ¿cómo cree Ud. que será esta adaptación?

   Muy Fácil ___ Fácil ___ Regular ___ Difícil ___ Muy Difícil __

A continuación se presenta un resumen de los indicadores a evaluar:

Variable	Indicadores	Items
Modelo de comunicación para Sistemas Multiagente	Implementable Adaptable	4,5 6,7
Utilización de los Web Services	Claro	1,2,3

Los instrumentos son mutuamente dependientes, es por esa razón que se utilizan los dos para evaluar cada uno de los indicadores.

Los materiales que se usarán son papel, material bibliográfico que podrá ser adquirido o consultado en bibliotecas y/o Internet, y todo tipo de suministros de computadoras como discos magnéticos, cintas de impresora, tinta, etc.

3.2. CAMPO DE VERIFICACIÓN

3.2.1. Ubicación espacial

El presente trabajo se llevará a cabo en la Ciudad de Arequipa y en el ciberespacio.

3.2.2. Ubicación temporal

El modelo se desarrollará a cabo el año 2008.

3.2.3. Unidades de estudio

Las unidades de estudio que se necesitan para poder elaborar el proyecto son el Modelo de Comunicación para Sistemas Multiagente utilizando Web Services y el simulador que se desarrollará.

Determinación cualitativa

Se trabajará con los miembros de una comunidad de desarrolladores de sistemas multiagente de la Pontificia Universidad Católica de Chile (16 miembros aproximadamente), quienes son dirigidos por la Dra. Rosa Alarcón. Estas personas se dedican a la investigación en la línea de metodologías multiagente y por lo tanto conocen sobre modelos de comunicación.

Determinación cuantitativa

Debido al grupo pequeño de personas con las que se tiene contacto, se toma la determinación de trabajar con todo el universo.

3.3. Estrategia de recolección de datos

La recolección de los datos será de tipo bibliográfico y de campo, la cual será denominada MOCOSMA que significa Modelo de Comunicación de Sistemas Multiagente.

Para poder obtener los datos de las unidades de estudio se buscará el contacto vía Internet y también por este medio se enviará el Cuestionario con el Modelo para que sea evaluado. Al momento de la recepción de los cuestionarios se procederá a sistematizar los resultados en Excel.

Una vez sistematizados, se aplicarán técnicas estadísticas de tabulación, obteniendo los gráficos correspondientes para su correspondiente análisis.

Los recursos necesarios para el desarrollo del trabajo de investigación son: Tiempo de los miembros de la comunidad de Internet, una computadora personal, suministros para la computadora, y papel.

IV. CRONOGRAMA DE TRABAJO

El trabajo será desarrollado de la siguiente manera:

Tiempo / Tareas	Diciembre				Enero				Febrero				Marzo				Abril				Mayo			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Determinación del tema	X	X																						
Antecedentes investigativos	X	X																						
Elaboración del proyecto		X	X	X																				
Presentación del proyecto				X																				
Observaciones modelo					X	X	X	X																
Revisión de modelos							X	X	X	X														
Revisión web services									X	X	X	X												
Desarrollo del modelo													X	X	X									
Desarrollo del simulador																X	X	X						
Recolección de datos (instrumento)																		X	X	X				
Estructuración de resultados																					X	X		
Informe final																							X	

V. BIBLIOGRAFÍA

[EURESCOM, 2000] EURESCOM, "MESSAGE: Methodology for Engineering Systems of Software Agents", Artículo técnico, 2000

[FININ, 1995] Finin, T. – Labrou, Y. – Mayfield, J., "KQML as an agent communication language", Universidad de Ciencias de la Computación e Ingeniería Eléctrica, USA, 1995.

[IGLESIAS, 1998] IGLESIAS, Carlos, "Definición De Una Metodología Para El Desarrollo De Sistemas Multiagente", Universidad Politécnica de Madrid, España, 1998.

[JENNINGS, 1998] JENNINGS, Nicholas, "A Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems", Volume 1, No. 1, 1998.

[MSDN, 2008] MSDN, "Servicios Web XML", http://www.microsoft.com/spanish/MSDN/estudiantes/desarrollo/avanzada/webservices.mspx, 2008

[RUSSEL, 1996] RUSSELL, Stuart, "Inteligencia Artificial. Un enfoque moderno", Prentice Hall Hispanoamericana, México, 1996

[STONE, 1997] Stone, P., Veloso, M., "Multiagent Systems: A Survey from a Machine Learning Perspective", http://www-2.cs.cmu.edu/afs/cs/usr/pstone/public/papers/97MAS-survey/revised-survey.html, 1997

[VAZQUEZ, 2004] Vázquez, Alberto, "Knowledge in Multiagent Systems", IJCAI 3.0 (Internet Junkie's Computer Artificial Intelligence), http://www.ijcai-03.org/~avazquez/knowledge/, 2004

[WIKIPEDIA, 2008] Wikipedia, "Servicio Web", http://es.wikipedia.org/wiki/Servicio_web, 2008

[WOOD, 2000] Wood, M. - DeLoach, S., "An Overview of the Multiagent Systems Engineering Methodology", Air Force Institute of Technology, USA, 2000.

[WOOLDRIDGE, 2000] WOOLDRIDGE, M. – Jennings, N. – Kinny, D., "The Gaia Methodology for Agent-Oriented Analysis and Design", Conferencia de Agentes Autónomos y Sistemas multiagnte, Holanda, 2000

Anexo 2: Evaluación del modelo de comunicación

1. ¿CÓMO CONSIDERA UD. EL NIVEL DE ENTENDIMIENTO DEL MODELO DE COMUNICACIÓN?

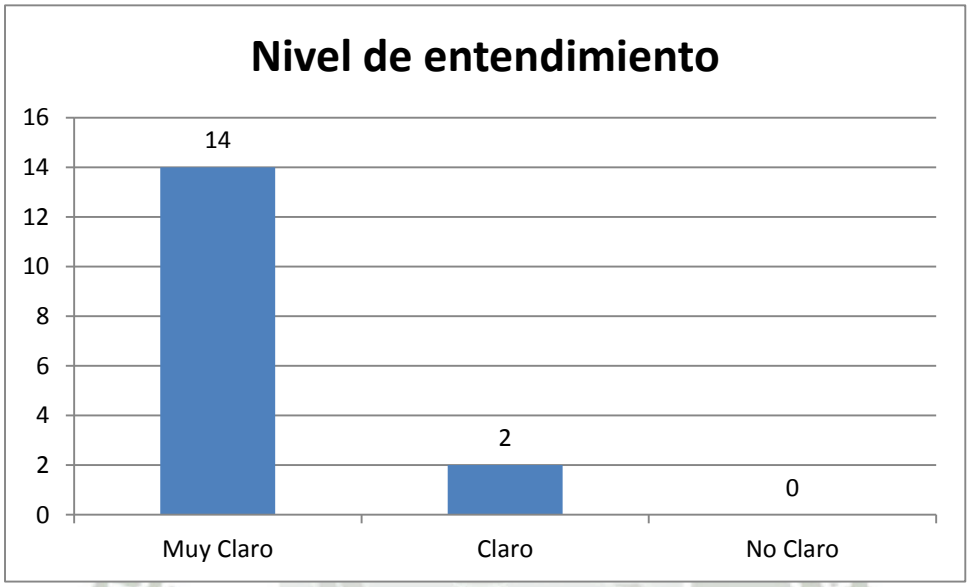


Figura 29: *MOCOSMA pregunta 1.*
Fuente: *Elaboración propia.*

Se puede apreciar que el 100% de los encuestados pudo entender el modelo de comunicación propuesto, de los cuales el 88% lo entendió muy claramente, y el restante 12% lo entendió de forma clara. Ninguno de los encuestados quedó con dudas luego de haber estudiado el modelo de comunicación propuesto.

Esto nos permite afirmar que los usuarios de esta propuesta no tendrán dificultad en entenderlo y por ende poder aplicarlo.

2. DESPUÉS DE UTILIZAR EL SIMULADOR, ¿LE QUEDÓ CLARO EL MODELO?

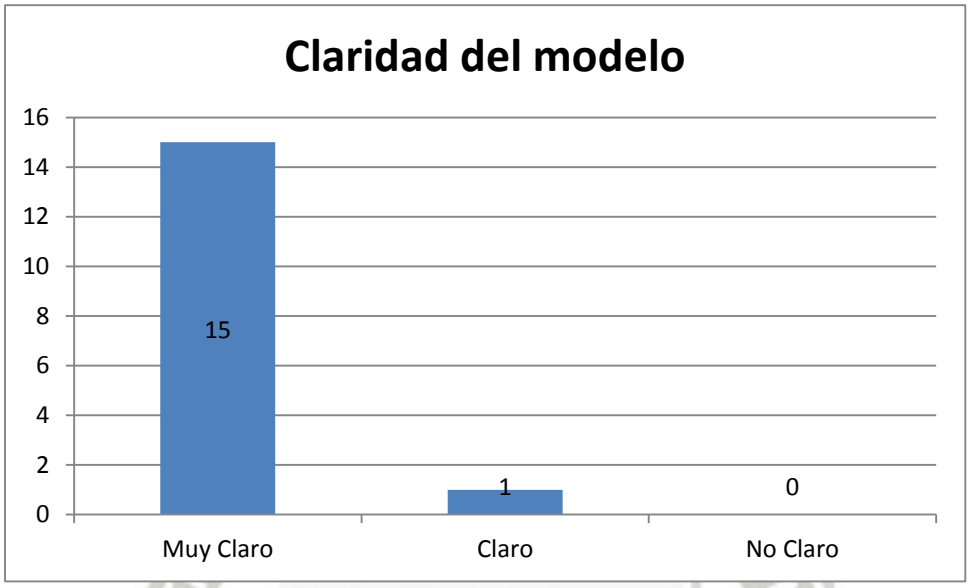


Figura 30: *MOCOSMA pregunta 2.*
Fuente: *Elaboración propia.*

Luego de haber utilizado el simulador el 100% de encuestados entendió el modelo de comunicación propuesto. De éstos, el 94% lo entendió claramente y sólo el 6% lo entendió de forma clara. Ninguno quedó con dudas sobre el modelo luego de utilizar el simulador.

De lo anterior podemos concluir que el simulador es una buena herramienta para mejorar el entendimiento del modelo de comunicación.

3. DESDE SU PERSPECTIVA, ¿ES SENCILLO EL MODELO PLANTEADO?

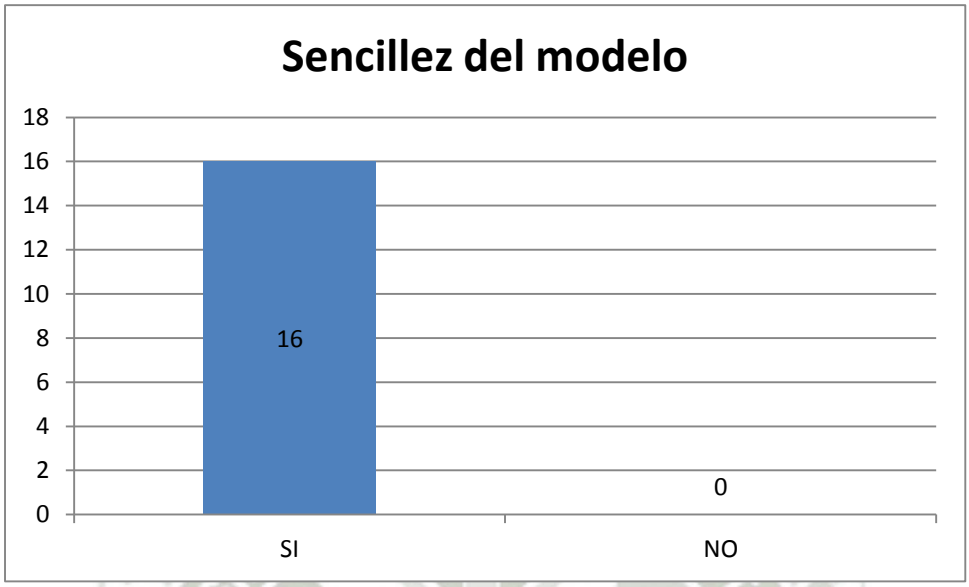


Figura 31: *MOCOSMA pregunta 3.*
Fuente: *Elaboración propia.*

El modelo de comunicación propuesto es sencillo, el 100% de los encuestados confirma esta afirmación. Esto nos permite adelantar una conclusión, la sencillez del modelo hace que su utilización sea un hecho y además, que sea factible su aplicabilidad.

4. BASÁNDOSE EN SU EXPERIENCIA, ¿CÓMO CONSIDERA UD. LA IMPLEMENTACIÓN DEL MODELO DE COMUNICACIÓN?

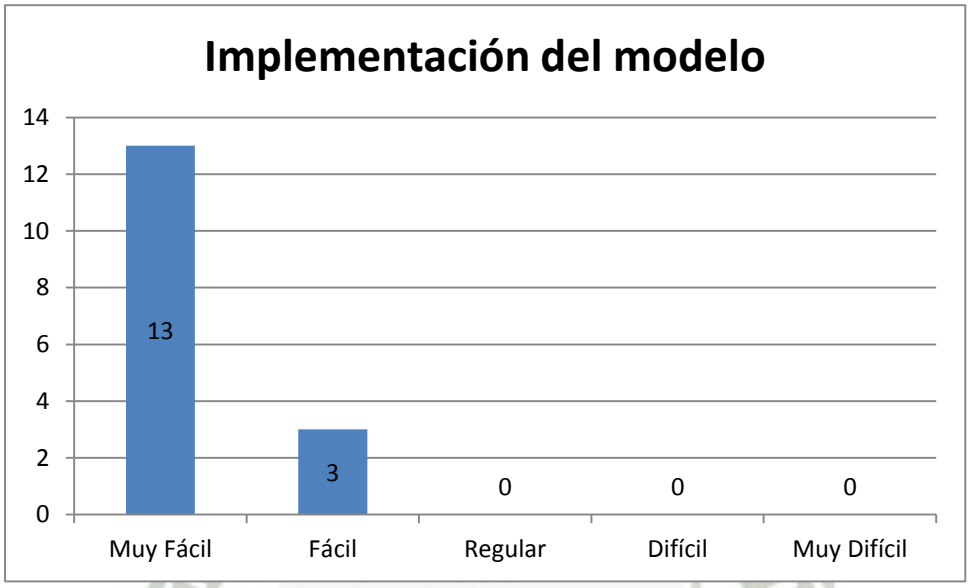


Figura 32: *MOCOSMA pregunta 4.*
Fuente: *Elaboración propia.*

El 100% de los encuestados afirma que la implementación del modelo de comunicación propuesto es fácil, de los cuales el 81% indica que esta implementación es muy sencilla. El 19% restante indica que es fácil la implementación. Podemos apreciar que ninguno de los encuestados encontró dificultad en la implementación del modelo.

Esta respuesta nos permitirá estar tranquilos al momento de liberar el modelo, ya que los lineamientos a seguir son claros y permiten su utilización, sin problemas para cualquier tipo de usuario.

5. ¿CONSIDERA UD. QUE EL MODELO TIENE TODO LO NECESARIO PARA COMPLEMENTAR LA IMPLEMENTACIÓN DE UN SISTEMA MULTIAGENTE?

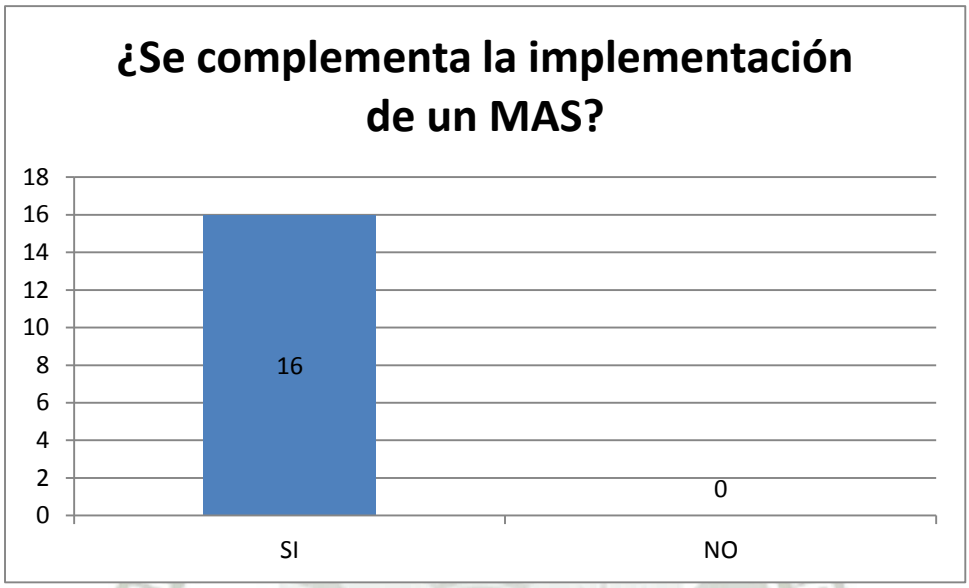


Figura 33: *MOCOSMA pregunta 5.*
Fuente: *Elaboración propia*

El 100% de los encuestados considera que el modelo de comunicación propuesto se inserta totalmente en un Sistema Multiagente y que además lo complementa adecuadamente. Esto nos permite afirmar que su integración a los demás modelos del Sistema Multiagente será de forma transparente y no generará ningún tipo de complicación.

6. ¿CREE UD. QUE EL MODELO SE ADAPTE Y SEA ENTENDIDO POR TODO TIPO DE USUARIO?

Figura 34: *MOCOSMA pregunta 6.*
Fuente: *Elaboración propia.*

El 100% de los encuestados afirma que no existirá ningún problema cuando usuarios con experiencia o novatos en Sistemas Multiagente utilicen el modelo de comunicación propuesto.

Esto permite afirmar que no será un modelo solamente teórico y dejado en el olvido, sino que podremos ver su rápida utilización en futuros Sistemas Multiagente. Además, sirve como base para adecuarlo y hacerlo evolucionar en otras casuísticas que surjan de su utilización.

7. SI SU RESPUESTA ANTERIOR FUE SI, RESPONDA LA PREGUNTA ¿CÓMO CREE UD. QUE SERÁ ESTA ADAPTACIÓN?

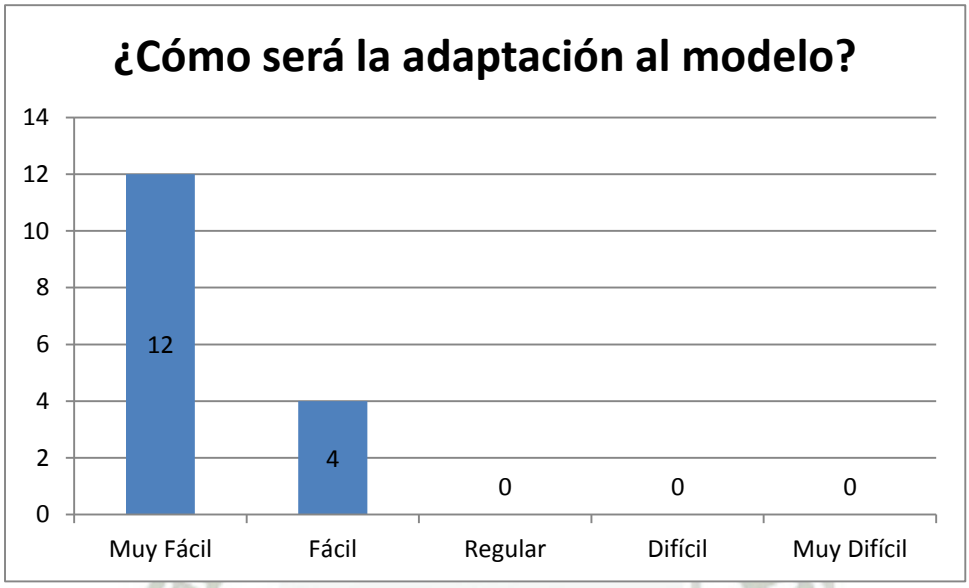


Figura 35: *MOCOSMA pregunta 7.*
Fuente: *Elaboración propia.*

Del 100% de los encuestado, el 75% afirma que la adaptación de los usuarios –con experiencia o novatos – será muy fácil, mientras que el 25% restante afirma que solamente será fácil, en caso los usuarios sean bastante nuevos en la materia.

Ninguno de los encuestados dijo que la adaptación sería difícil, lo cual refuerza las conclusiones obtenidas en las preguntas anteriores.

Anexo 3: Guía de Funcionamiento del simulador

Al ejecutar el Simulador del Modelo de Comunicaciones (Pizarra), se muestra la siguiente ventana.

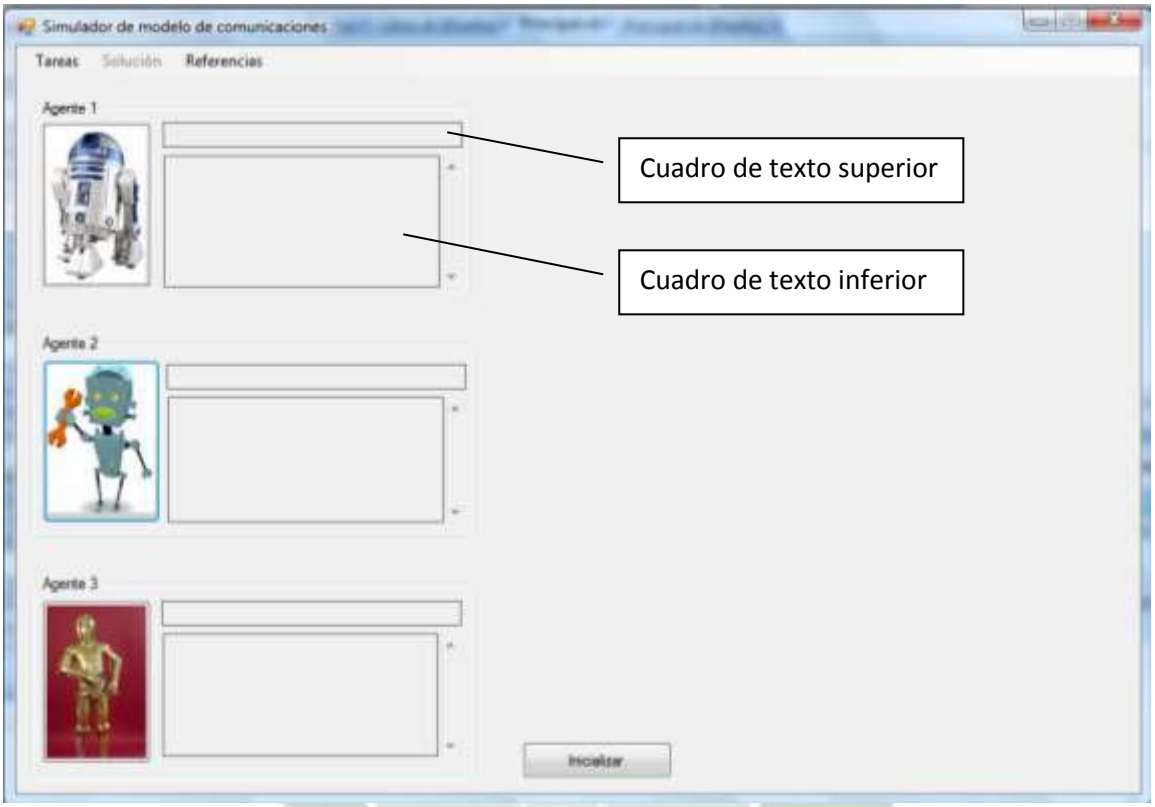


Figura 36: *Pantalla principal del simulador.*
Fuente: *Pizarra v 1.0*

En esta ventana se pueden apreciar tres secciones importantes: Barra de menús, sección de agentes y botón Inicializar.

Barra de menús

- *Tareas*: Presenta tres opciones: Elaborar Libro y Construir Casa para seleccionar y ejecutar un proceso. La tercera opción es Salir.
- *Solución*: Menú que se activa cuando ha finalizado un proceso, muestra en una nueva ventana la secuencia de ejecución del proceso.
- *Referencias*: Permite ver la información detallada de cada uno de los procesos. Como ejemplo tenemos la figura A1-2.

Sección de agentes

- *Cuadro de texto superior*: Cuando se esté ejecutando un proceso, mostrará el nuevo conocimiento adquirido por el agente.

- *Cuadro de texto inferior*: Cuando se esté ejecutando un proceso, mostrará el conocimiento acumulado del agente.

Botón Inicializar

Permite inicializar la base de datos, para poder ejecutar nuevamente los procesos.

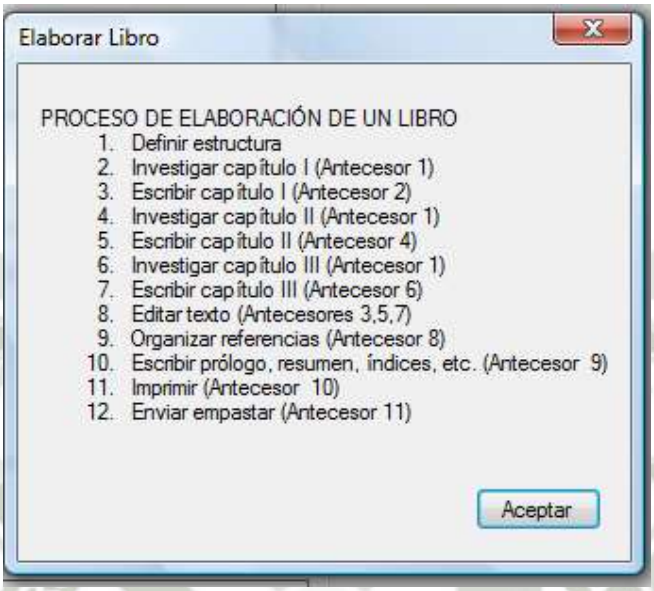


Figura 37: *Información del Proceso de elaboración de un libro, que muestra las actividades asociadas y sus respectivas restricciones.*
Fuente: *Pizarra v 1.0*

Una vez que se seleccione alguna de las opciones del menú Tareas (i.e., Elaborar Libro, Construir Casa), el proceso se comienza a ejecutar. Al ejecutar el proceso se pueden apreciar las siguientes características (las cuales se pueden apreciar en la figura A1-3):

- Se muestra el Agente activo, el cual ha realizado una consulta y se le ha asignado una tarea. El agente activo se muestra en color celeste.
- Se puede apreciar el nuevo conocimiento de cada agente (cuadro de texto superior), el cual se muestra además en color rojo.
- Se puede apreciar el conocimiento acumulado de los agentes (cuadro de texto inferior). En el cual se colocan las siguientes etiquetas:
    - Inicio: Señala que el proceso inicia con ese agente.
    - Consulta: Señala que ese agente realizó una consulta para saber si existía alguna actividad libre.
    - Tarea: Indica que ese agente ejecutó esa tarea.
    - Tarea "etiquetada": Indica qué agente ejecutó esa tarea, e.g., A1::Elaborar Planos. El agente 1 elaboró los planos.
- Se puede ver el seguimiento de las actividades, así como su estado en la grilla que aparece en el lado derecho de la ventana.

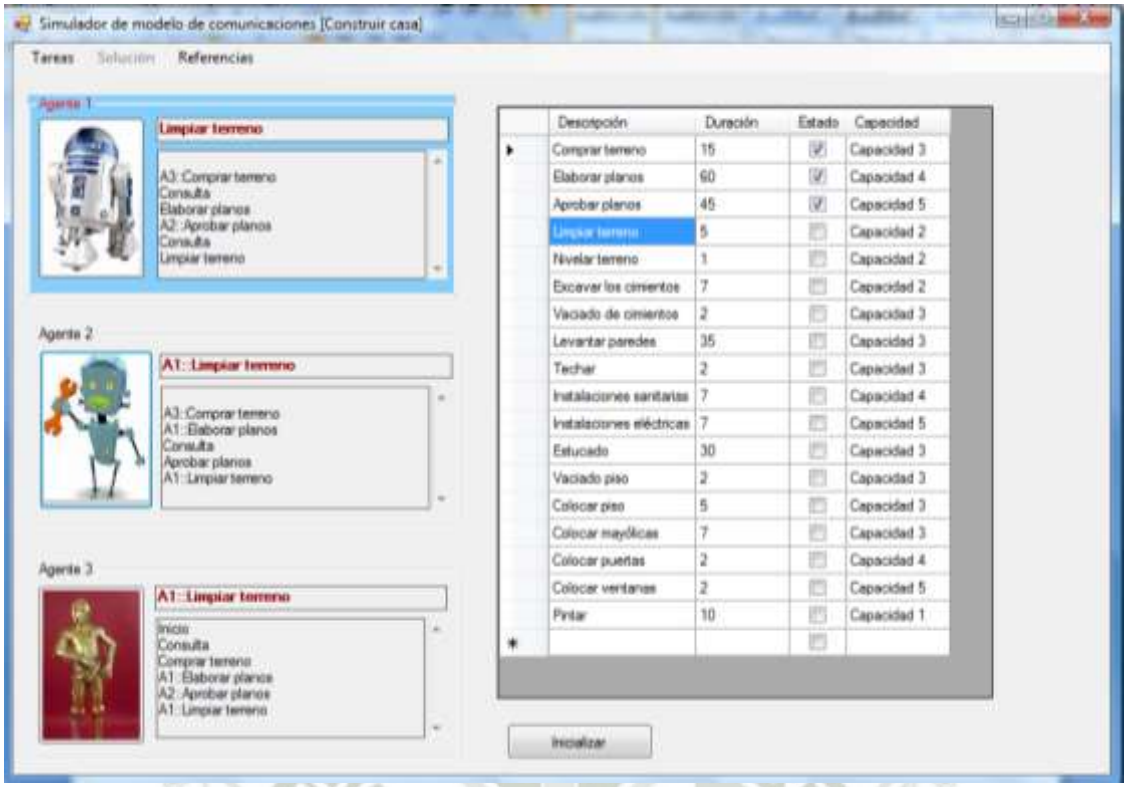


Figura 38: *Ejecución del proceso Construir Casa.*
Fuente: *Pizarra v 1.0*

Al finalizar la ejecución del proceso el simulador avisará mediante una ventana de mensaje, como la que se muestra en la figura A1-4.

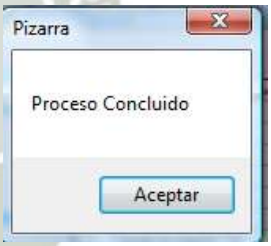


Figura 39: *Mensaje.*
Fuente: *Pizarra v 1.0*

Se puede hacer el seguimiento de la ejecución de las tareas al leer el conocimiento acumulado de cualquiera de los agentes (preferentemente el que tenga la etiqueta *Inicio*). Por ejemplo, en la figura A1-5 se puede apreciar el conocimiento acumulado del agente 3. En esta figura se puede apreciar que este agente inició el proceso, realizó una consulta y se le asignó la tarea *Comprar terreno*. Luego el agente 1 elaboró los planos, el agente 3 aprobó dichos planos, etc.

Figura 40: *Seguimiento de la ejecución del proceso.*
Fuente: *Pizarra v 1.0*

Si la lectura anterior no es muy clara, se puede seleccionar el menú *Solución* (que para este momento ya esta activado), y se mostrará la ejecución en un formato más sencillo de leer. Un ejemplo de solución se puede apreciar en la figura A1-6.

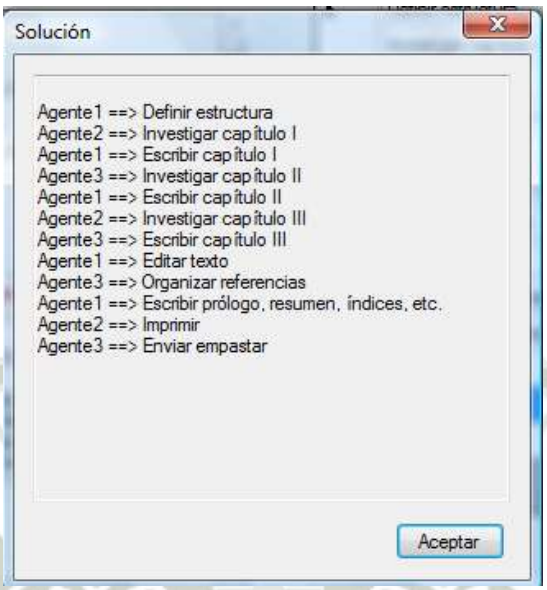


Figura 41: *Solución del proceso de elaboración de un libro.*
Fuente: *Pizarra v 1.0*

Anexo 4: Código de implementación del simulador

El código de implementación y una breve descripción del mismo se muestran en este anexo.

Servicio Web Informador

Servicio web que se comunica con la base de datos para verificar si existe alguna tarea libre en el proceso active. Retorna 0 si el proceso a concluido o el ID de la tarea.

```
Public Class Service1
    Inherits System.Web.Services.WebService
    Dim xStringConex As String = "<<Cadena de conexión>>"
    Dim objSqlConnection As New SqlConnection
    Dim objSqlCommand As New SqlCommand
    <WebMethod()> _
    Public Function BuscaTarea(ByVal proceso As String, ByVal agente As
Integer) As Integer
        Dim rTarea As Integer
        Dim proc As String
        If proceso = "L" Then
            proc = 1
        Else
            proc = 2
        End If
        objSqlConnection.ConnectionString = xStringConex
        objSqlCommand.CommandType = CommandType.StoredProcedure
        objSqlCommand.CommandText = "sp_Selecciona_Tarea"
        objSqlCommand.Parameters.Add("@Agente", Data.SqlDbType.BigInt).Value =
agente
        objSqlCommand.Parameters.Add("@Proceso", Data.SqlDbType.BigInt).Value
= proc
        objSqlCommand.Parameters.Add("@Tarea", SqlDbType.NVarChar,
50).Direction = ParameterDirection.Output
        objSqlCommand.Connection = objSqlConnection
        objSqlConnection.Open()
        objSqlCommand.ExecuteScalar()
        rTarea = objSqlCommand.Parameters(2).Value
        objSqlConnection.Close()
        Return rTarea
    End Function
End Class
```

Servicio Web Almacenador

Servicio web que se comunica con la base de datos y retorna la última actividad seleccionada para asignarla a un agente.

```
Public Class Service1
    Inherits System.Web.Services.WebService
    Dim xStringConex As String = "<<Cadena de conexión>>"
    Dim objSqlConnection As New SqlConnection
    Dim objSqlCommand As New SqlCommand
    <WebMethod()> _
    Public Function Base_Conocimiento() As String
        Dim rTarea As String
        objSqlConnection.ConnectionString = xStringConex
```

```
        objSqlCommand.CommandType = CommandType.StoredProcedure
        objSqlCommand.CommandText = "sp_Devuelve_Tarea"
        objSqlCommand.Parameters.Add("@Tarea", SqlDbType.NVarChar,
50).Direction = ParameterDirection.Output
        objSqlCommand.Connection = objSqlConnection
        objSqlConnection.Open()
        objSqlCommand.ExecuteScalar()
        rTarea = objSqlCommand.Parameters(0).Value
        objSqlConnection.Close()
        Return rTarea
    End Function
End Class
```

sp_Selecciona_Tarea

Procedimiento almacenado que busca tareas disponibles en el proceso activo, en caso que el proceso ya esté concluido retorna 0. En caso contrario selecciona la actividad, le cambia su estado y almacena la relación Agente-Tarea.

```
CREATE PROCEDURE [dbo].[sp_Selecciona_Tarea]
      @Agente int,
      @Proceso int,
      @Tarea bigint OUTPUT
AS
BEGIN
      SET NOCOUNT ON;
      DECLARE @trID as bigint
      DECLARE @trEstado as bit
      DECLARE @anAntecesor as bigint
      SET @Tarea = 0 --Para indicar que proceso a finalizado
            DECLARE cursorTareas CURSOR FOR
            SELECT trID, trEStado
            FROM Tarea
            WHERE trProceso = @Proceso
      OPEN cursorTareas
            FETCH NEXT FROM cursorTareas INTO
            @trID,
            @trEstado
      WHILE @@fetch_status = 0
      BEGIN
            IF @trEstado = 0
            BEGIN
                  SET @Tarea = @trID
                  UPDATE Tarea SET trEstado = 1 WHERE trID = @trID
                  INSERT INTO AgenteTarea VALUES (@Agente, @trID, GETDATE())
                  BREAK
            END
            FETCH NEXT FROM cursorTareas INTO
                  @trID,
                  @trEstado
      END
      CLOSE cursorTareas
      DEALLOCATE cursorTareas
      IF @Tarea = 0
      BEGIN
            UPDATE Proceso SET prEstado = 1 WHERE prID = @Proceso
      END
END
```

sp_Devuelve_Tarea

Procedimiento almacenado que busca y retorna la última actividad seleccionada, la cual se agregará como conocimiento a cada uno de los agentes.

```
CREATE PROCEDURE [dbo].[sp_Devuelve_Tarea]
	@Tarea nvarchar(50) OUTPUT
AS
BEGIN
	SET NOCOUNT ON;

	DECLARE @atTarea integer

	SELECT TOP 1 @atTarea = atTarea
	FROM AgenteTarea
	ORDER BY atFecha DESC

	SELECT @Tarea = trDescripcion
	FROM Tarea
	WHERE trID = @atTarea
END
```

sp_Inicializar

Procedimiento almacenado que inicializa la base datos para una nueva ejecución.

```
CREATE PROCEDURE [dbo].[sp_Inicializar]
AS
BEGIN
	SET NOCOUNT ON;
	TRUNCATE TABLE AgenteTarea
	UPDATE Tarea SET trEstado = 0
	UPDATE Proceso SET prEstado = 0
END
```

sp_TareasLibro_Completo

Procedimiento almacenado que retorna todas las tarea del proceso Elaborar libro.

```
CREATE PROCEDURE [dbo].[sp_TareasLibro_Completo]
AS
BEGIN
	SET NOCOUNT ON;
	SELECT trDescripcion as 'Descripción', trDuracion as 'Duración',
trEstado as 'Estado', caDescripcion as 'Capacidad'
	FROM Tarea, Capacidad
	WHERE trProceso = 1 and (trCapacidad =caID)
END
```

sp_TareasCasa_Completo

Procedimiento almacenado que retorna todas las tarea del proceso Construir casa.

```
CREATE PROCEDURE [dbo].[sp_TareasCasa_Completo]
AS
BEGIN
	SELECT trDescripcion as 'Descripción', trDuracion as 'Duración',
trEstado as 'Estado', caDescripcion as 'Capacidad'
	FROM Tarea, Capacidad
	WHERE trProceso = 2 and (trCapacidad =caID)
END
```

Llamar proceso

Procedimiento que se conecta a la base datos para obtener todas las tareas del proceso activo, con este resultado llena la grilla de seguimiento de tareas.

```
Private Sub Llamar_Proceso(ByVal Titulo_Adicional, ByVal
Procedimiento_Almacenado)
   Dim xStringConex As String = "<<Cadena de conexión>>"
   Dim objSqlConnection As New SqlConnection
   Dim objSqlCommand As New SqlCommand

   SoluciónToolStripMenuItem.Enabled = False
   Me.Text = Titulo + Titulo_Adicional
   DGVPizarra.DataSource = Nothing
   Try
       Dim ds As DataSet = New DataSet()
       objSqlConnection.ConnectionString = xStringConex

       Dim da As SqlDataAdapter = New SqlDataAdapter(Procedimiento_Almacenado,
objSqlConnection)
       da.SelectCommand.CommandType = CommandType.StoredProcedure

       objSqlConnection.Open()
       da.Fill(ds, "DataSet1")

       DGVPizarra.Visible = True
       DGVPizarra.AutoSize = True
       DGVPizarra.DataSource = ds
       DGVPizarra.DataMember = "DataSet1"
       DGVPizarra.Refresh()
       objSqlConnection.Close()

   Catch ex As Exception
       Label1.Text = ex.Message
   End Try
End Sub
```

Seleccionar

Procedimiento que simula la llegada y consulta de un agente, llama a los servicios web correspondientes para buscar tareas libres y asignarlas a los agentes correspondientes.

```
Private Sub Seleccionar(ByVal Proceso)
   Dim Agente_Random As Integer = 0
   Dim Agente_Anterior As Integer = 0
   Dim Tarea_Realizar As String
   Dim Busca As Integer = 1
   Dim x As Integer = 0

   Inicio = "0"
   Sol = ""
   While Busca <> 0
      While True
          Randomize()
          Agente_Random = CInt(Int((3 * Rnd()) + 1))
          If Agente_Libre(Agente_Random) = 1 Then
              Desocupar_Agentes(Agente_Anterior)
              Exit While
          End If
      End While
      Agente_Anterior = Agente_Random

      Busca = objWS_Informador.BuscaTarea(Proceso, Agente_Random)
      If Busca = 0 Then
          MsgBox("Proceso Concluido")
```

```
            Exit While
        End If
        Tarea_Realizar = objWS_Almacenador.Base_Conocimiento
        DGVPizarra.Rows(x).Cells(0).Selected = True
        If x <> 0 Then
            DGVPizarra.Rows(x - 1).Cells(0).Selected = False
        End If
        Agregar_Conocimiento(Agente_Random, Tarea_Realizar)
        Sol = Sol + vbCrLf + "Agente" + Trim(Str(Agente_Random)) + " ==> " +
Tarea_Realizar

        Cambia_Color(Agente_Random)
        DGVPizarra.Rows(x).Cells(2).Value = 1

        x = x + 1
    End While
    SoluciónToolStripMenuItem.Enabled = True
End Sub
```

Anexo 5: Tabulación de resultados de las encuestas

Es este anexo se muestran las hojas de tabulación de los resultados del cuestionario aplicado al grupo de expertos. Es bueno aclarar que estas matrices son la base de los resultados mostrados en el Anexo 2.

Pregunta 1: ¿Cómo considera Ud. el nivel de entendimiento del modelo de comunicación?

Muy Claro	14
Claro	2
No Claro	0
	16

Pregunta 2: Después de utilizar el simulador, ¿le quedó claro el modelo?

Muy Claro	15
Claro	1
No Claro	0
	16

Pregunta 3: Desde su perspectiva, ¿es sencillo el modelo planteado?

SI	16
NO	0
	16

Pregunta 4: Basándose en su experiencia, ¿cómo considera Ud. la implementación del modelo de comunicación?

Muy Fácil	13
Fácil	3
Regular	0
Difícil	0
Muy Difícil	0
	16

Pregunta 5: ¿Considera Ud. que el modelo tiene todo lo necesario para complementar la implementación de un sistema multiagente?

SI	16
NO	0
	16

Pregunta 6: ¿Cree Ud. que el modelo se adapte y sea entendido por todo tipo de usuario?

SI	16
NO	0
	16

Pregunta 7: Si su respuesta anterior fue SI, responda la pregunta ¿cómo cree Ud. que será esta adaptación?

Muy Fácil	12
Fácil	4
Regular	0
Difícil	0
Muy Difícil	0
	16