UNIVERSIDAD CENTROCCIDENTAL
"LISANDRO ALVARADO"

DISEÑO DE UN SISTEMA MULTIAGENTE PARA LA GESTIÓN DE SERVICIOS DE EMERGENCIA MÉDICA HOSPITALARIA USANDO METODOLOGÍA INGENIAS

MARITZA VALENTINA BIELIUKAS DORANTE

Barquisimeto, 2010

UNIVERSIDAD CENTROCCIDENTAL "LISANDRO ALVARADO"
DECANATO DE CIENCIAS Y TECNOLOGÍA
MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

.

DISEÑO DE UN SISTEMA MULTIAGENTE PARA LA GESTIÓN DE SERVICIOS DE EMERGENCIA MÉDICA HOSPITALARIA USANDO METODOLOGÍA INGENIAS

Trabajo para optar al título de
Magíster Scientiarum en Ciencias de la Computación

AUTORA: MARITZA VALENTINA BIELIUKAS DORANTE
TUTORA: MARÍA AUXILIADORA PÉREZ

Barquisimeto, 2010

DEDICATORIA

A Dios Todopoderoso, creador del aire que respiro y del amanecer que a diario admiro, guía inseparable de mis días; luz que ilumina mi camino.

A la Divina Pastora, patrona de la tierra en que nací, que con tu bendición llenas cada momento de mi vida para luchar por cada uno de los sueños que quiero lograr.

A mi Madre, mujer maravillosa, mi motivo para vivir, mi mayor orgullo y mi mejor ejemplo de lucha y de salir adelante a pesar de los obstáculos que se presentan en el camino…. **Te Amo.**

A mi Hermana Ybelisse, por estar conmigo siempre, por confiar y apoyarme en todo momento. **Te Amo**.

A mi hermana Ileana, por ir de la mano conmigo en todo momento, por ser a veces mi hermanita menor y en otros momentos la hermana mayor que me regaña y protege, pero siempre juntas y por confiar y apoyarme para que yo salga adelante . **Te Amo**

A mis tres angelitos Santy, Sebas y Salomón, mis sobrinos hermosos, razones maravillosas que llenan mi vida de plenitud y entusiasmo, son fuentes de alegría y esperanza de cada día.

A mi Tutora, por el apoyo incondicional en el desarrollo de este proyecto y por el cariño que me ha brindado.

AGRADECIMIENTO

A mis abuelos (Enma y Oscar), por darme su apoyo, y por brindarme sus sabios consejos…. Los Amo.

A mi gran casa de estudio UCLA, mi casa, parte de mi vida, a la que le debo muchas cosas, mi orgullo. Lugar donde crecí, donde coseche triunfos, llore, rei, conocí a mis mejores amigos. Agradezco a cada docente, cada administrativo, cada obrero, en fin a cada persona que forma parte de esta casa, que me ha permitido desenvolverme como ser humano, como profesional, donde obtuve conocimientos para lograr todo lo que quiera alcanzar.

A la Profesora Haidee, mas que una amiga, es parte de mi familia, por estar siempre pendiente de mí, apoyándome y dándome mucho animo para salir adelante, demostrándome su cariño y brindándome siempre el mejor de los consejos.

A mi mejor amiga Marialberth, por estar siempre conmigo y apoyándome con su gran amistad.

A mi gran amigo Pedro, por ser tan especial e incondicional conmigo.

A la profesora Carme Teresa, por su gran amistad, por hacerme sentir su cariño en los peores y mejores momentos de mi vida.

Al Dr. Marco Tulio Mendoza, hombre de gran valor para mi, ya que en el logre escuchar sus sabios consejos de un padre a una hija y hacerme reír cuando le llegaba con lagrimas en mis ojos

A mi amigo Jesús por ir de la mano conmigo en el recorrido de toda la maestría y darme apoyo en el desarrollo de este proyecto.

A mi amigo Juan, por brindarme sus conocimientos y su apoyo para lograr la culminación de esta meta.

A mis grandes amigos, (Jennifer, Ítalo, Adolfo, Mariana, Mery, Ciriant, Angélica y David) personas especiales que me brindan su apoyo y me motivan a seguir adelante con mis metas.

A mis profesores (Maritza, María, Belkis, José Gregorio, Isacura, Margarita, Virginia) por brindarme los conocimientos necesarios para poder alcanzar esta meta.

Al Dr. Antonio Franco, coordinador del servicio de emergencia del Hospital Central Universitario "Antonio María Pineda", por su colaboración en el desarrollo de este proyecto.

Al doctor y amigo Eleazar Graterón, por brindarme su apoyo en cuanto al levantamiento de información sobre el funcionamiento de la emergencia del hospital, la cual ayudo con el desarrollo del planteamiento del problema**.**

A las enfermeras, secretarias y al personal que labora en el servicio de emergencia del Hospital Central Universitario "Antonio María Pineda", por brindarme la información necesaria para realizar el planteamiento del problema.

Gracias a todos los que junto a mi transitaron por este camino lleno de momentos dulces, amargos, triunfos y derrotas pero que me dejaron una gran enseñaza.

INDICE GENERAL

INDICE DE CUADROS

CUADRO **Pág.**

INDICE DE FIGURAS

FIGURAS **Pág.**

UNIVERSIDAD CENTROCCIDENTAL LISANDRO ALVARADO"

DECANATO DE CIENCIAS Y TECNOLOGÍA

MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN

DISEÑO DE UN SISTEMA MULTIAGENTE PARA LA GESTIÓN DE SERVICIOS DE EMERGENCIA MÉDICA HOSPITALARIA USANDO METODOLOGÍA INGENIAS

Autora: Maritza Valentina Bieliukas Dorante

Tutora: María Auxiliadora Pérez

RESUMEN

La presente investigación se ha desarrollado en el servicio de emergencia del Hospital Central Universitario "Dr. Antonio María Pineda", ubicado en la ciudad de Barquisimeto – Estado Lara – Venezuela, dedicado a la atención primaria de los pacientes, ofreciendo un tratamiento inicial a los pacientes que presentan un amplia variabilidad de lesiones y enfermedades que pueden ser mortales y que requieren atención inmediata. La misma tiene como finalidad presentar el desarrollo de un prototipo de un Sistema Multiagente para la gestión de servicios de emergencia médica hospitalaria usando metodología INGENIAS. Este trabajo esta enmarcado en la modalidad de proyecto factible y apoyado en una investigación documental y de campo. Para la obtención de los datos se realizaron entrevistas a diferentes personas que laboran en el Servicio de Emergencia de dicho hospital. El análisis de la información obtenida del levantamiento de información realizado, permitió realizar el diagnóstico de la situación actual del servicio, así como analizar la complejidad de los procesos que allí ocurren. El desarrollo de un prototipo del sistema multiagente, permitirá abordar y tratar el problema planteado, el cual se basa en los diferentes factores que dificultan la gestión del servicio de emergencia, aprovechando la interacción autónoma de agentes artificiales entre ellos y con humanos. De esta manera se establece un enlace con los principios fundamentales de la inteligencia artificial. El prototipo fue diseñado con la herramienta de modelado de agente INGENIAS Development Kit IDK y en un ambiente de programación Java, con librerías de Jade para simular la cooperación y la interacción entre los agentes. El principal aporte de esta investigación es que se logró simular el funcionamiento de todos los procesos que ocurren dentro del servicio de emergencia, en particular, las pruebas permitieron verificar la optimización del proceso de recepción de los pacientes, que es uno de los procesos que presenta en la actualidad mayor dificultad.

Descriptores: Gestión de Servicios de Emergencia, Prototipo, Sistemas Multiagentes, Metodología INGENIAS, IDK, Java, Jade.

INTRODUCCIÓN

El paradigma de agentes y sistemas multiagente constituye actualmente un área de creciente interés dentro de la Inteligencia Artificial, entre otras razones, por ser aplicable a la resolución de problemas complejos no resueltos de manera satisfactoria mediante técnicas clásicas. según Jennings (1998), Numerosas aplicaciones basadas en este nuevo paradigma vienen ya siendo empleadas en infinidad de áreas entre las que se pueden mencionar, control de procesos, procesos de producción, control de tráfico aéreo, aplicaciones comerciales, gestión de información, comercio electrónico, aplicaciones médicas, juegos, etc.

En el área de investigación de agentes uno de los problemas que actualmente se enfrentan es que cada vez son necesarios nuevos métodos, técnicas y herramientas que faciliten el desarrollo de aplicaciones basadas en dicho paradigma. Así mismo, el progreso en la ingeniería del software, en los últimos años se ha realizado gracias al desarrollo de abstracciones más poderosas y naturales para modelar sistemas más complejos.

Por otra parte, la aceptación de métodos en la industria y/o empresa depende de la existencia de las herramientas necesarias que soporten el análisis, diseño e implementación de agentes de software. En los últimos años han surgido diferentes aproximaciones que tratan de presentar una metodología apropiada para el desarrollo de sistemas multiagente.

La gran variedad y disparidad de problemas a los que se enfrenta la inteligencia artificial, tiene en el concepto de agente inteligente, una robusta herramienta para el diseño de programas y de plataformas de programación distribuida. Los agentes inteligentes son ahora un concepto de diseño, de ingeniería de software que se

acompaña generalmente con metodologías de diseño basadas en utilidades y con plataformas de implementación específicas de cada metodología.

Es así como, los agentes inteligentes permiten identificar las características generales que adopta la solución de todo un conjunto de problemas, a la vez que facilitan la identificación de los mismos. En otras palabras, dado un problema a resolver de un dominio en particular, dicho problema se puede describir primeramente en función de sus características para posteriormente identificar la categoría a la que pertenece y así el tipo de agente que ofrece una solución al mismo.

Como un lógico proceso evolutivo, según Clemmer y Gardem (1995) la Medicina ha ido asimilando la introducción de las computadoras para agilizar y mejorar los procesos de apoyo médico, teniendo una gran influencia, que sigue aumentando más cada día con la introducción de la Inteligencia Artificial en la vigilancia del paciente con complejos equipos biomédicos, realización de procesamiento voluminoso de información para la toma de decisiones y muchas otras aplicaciones. Es así como el surgimiento de la Informática Médica, comprende una amplia gama de cuestiones de la organización y del uso de la información biomédica. El objetivo de la Informática Médica es reforzar y mejorar la toma de decisiones médicas y la atención al paciente.

El gran impacto que han producido los avances tecnológicos en la medicina, han permitido descubrir los cambios que ocurren en el organismo humano y las consecuencias del entorno. La tecnología y la medicina deben ir tomadas de la mano; en la actualidad se observa el desarrollo de innumerables equipos médicos que forman parte de los avances tecnológicos, cuya evolución ha venido incrementando a lo largo del tiempo.

La informática médica experimenta un gran crecimiento en la actualidad; en este campo surgen continuamente nuevas aplicaciones, por lo que no es extraño que las técnicas multiagentes también se estén introduciendo aquí, principalmente en el área de cuidado y atención de pacientes.

Por otra parte, en medicina, emergencia se refiere a todo problema médico – quirúrgico agudo, que ponga en peligro la vida, o la pérdida de un órgano o una función y que requiera atención inmediata. Es por eso que se considera de suma importancia para la sociedad contar con un servicio de emergencia, el cual es una unidad dedicada y comprometida a contribuir y mejorar la salud de la población, atendiendo y estabilizando de manera oportuna, a los pacientes en situaciones de emergencia y urgencia.

En la actualidad los Sistemas Multiagentes, han servido para dar soporte a las decisiones del médico durante su guardia, brindándole mejor información procesada de mayor calidad. En estos ambientes tan complejos, la probabilidad de ocurrencia del error humano se ve aumentada, debido a las condiciones desfavorables en las que deben desempeñarse estos profesionales: elevado estrés, escaso tiempo disponible para la toma de decisiones, gran demanda de pacientes, escasez de recursos humanos, físicos y de materiales, gran variabilidad de las situaciones, interacción entre diferentes grupos de profesionales, baja calidad de la información, pacientes que acuden al servicio con emergencias menores, entre otros casos.

Por lo tanto el uso de una herramienta informática que permita mejorar la calidad de la información en estos ambientes permitirá mejorar la atención de los pacientes y aumentar la calidad de trabajo de los profesionales de la medicina. Teniendo en cuenta la complejidad del problema y la necesidad de adaptabilidad a diferentes ámbitos de trabajo y utilizando además el desarrollo y avances de la inteligencia artificial, como ciencia que puede dar respuesta efectiva en diversas áreas del conocimiento, se plantea el diseño de un sistema multiagente para la gestión del servicio de emergencia médica hospitalaria usando metodología INGENIAS.

El trabajo está compuesto de los siguientes capítulos:

Capitulo I: El problema, es donde se realiza el planteamiento del problema y hacia donde va la investigación, de igual forma se describen los objetivos generales y específicos, así como también, alcance, limitaciones y la justificación de la misma.

Capítulo II: Marco teórico, en este es donde se plasman los antecedentes investigados y las bases teóricas que sirven de apoyo a la investigación.

Capitulo III: Marco metodológico, muestra el diseño de la investigación, la factibilidad, recursos económicos y técnicos, y el diseño de la propuesta.

Capitulo IV: Análisis de Resultados, muestra los resultados obtenidos en la investigación y la descripción de los detalles de diseño del prototipo del sistema multiagente.

Capitulo V: Conclusiones y recomendaciones, es donde se plantean las conclusiones a que llego el investigador luego de la culminación del proyecto, así como también, las posibles recomendaciones para mejorar el mismo.

CAPITULO I

EL PROBLEMA

Planteamiento del Problema

De acuerdo a la Sociedad Venezolana de Medicina de Emergencia y Desastre, en la presentación de las "Bases Legales de la Atención Prehospitalaria en Venezuela", se define Emergencia Médica como: "todas aquellas situaciones que afectan al ser humano y que se presentan en forma intempestiva, inesperada e inoportuna, como un accidente que sobreviene y que provoca una insuficiencia aguda de las funciones vitales de órganos, aparatos o sistemas y que requieren de la atención inmediata de un médico".

Según Villatoro (2005), el servicio de emergencias es una unidad integral que el hospital dedica para recibir, estabilizar y manejar pacientes que se presentan, solos o mediante referencia, con una gran variedad de condiciones de emergencias. Los cuidados a estos pacientes deben ser administrados con un alto estándar de calidad, dado que la comunidad percibe la necesidad de atención como aguda o urgente, incluyendo la admisión hospitalaria.

Los principales casos que se consideran emergencia para este tipo de servicio incluyen: disnea súbita (dificultad respiratoria repentina), hemorragias activas por cualquier vía, dolor torácico súbito y persistente, perdida del conocimiento sin recuperación espontánea, heridas extensas y/ o amputación, heridas por arma de fuego, heridas por arma blanca, crisis convulsivas, parálisis súbita de cualquier extremidad, ingestión de sustancias tóxicas, presencia de cuerpos extraños en vías

respiratorias o digestivas, retención aguda de orina, mordeduras o picadas de animales venenosos, quemaduras de primero, segundo y tercer grado.

El funcionamiento de este servicio, puede ser descrito de la siguiente manera:

- **Cuidados del Paciente:**

  Constituye la función principal de este servicio, cuenta para la prestación del mismo, con un área de recepción, un área de triage (método de selección y clasificación de pacientes basados en sus necesidades terapéuticas y los recursos disponibles para su atención, de acuerdo al Programa Avanzado de Apoyo Vital en Trauma para Médicos. (2005)), un área de valoración inicial y sala de observación. Además provee ó soluciona los cuidados posteriores de los pacientes atendidos más allá de la fase de emergencias, (ingreso a piso) dependiendo de la infraestructura hospitalaria.

- **Estructura:**

  Es una parte del hospital y cuenta con licencia de la autoridad territorial responsable. Debe estar diseñada con el propósito de atender emergencias y tener un área con la capacidad de soporte de vida avanzada incluyendo ventilación mecánica, para la recepción y estabilización de pacientes críticamente enfermos. Así mismo, debe poseer un área de descontaminación de pacientes y contar con el apoyo las veinticuatro (24) horas del día, de los servicios de: laboratorio, imagenología, patología y del servicio de microbiología por lo menos seis (6) días por semana en turnos de doce (12) horas.

- **Interconsultas:**

  Posee un directorio de especialistas ó contar con ellos para solicitar opinión o referencia las veinticuatro (24) horas del día, principalmente en las especialidades de anestesiología, cirugía general, ginecología y obstetricia, medicina interna, ortopedia, cuidados intensivos, entre otras. Cuenta con capacidad para la atención y valoración en angiología, infectología, neurocirugía, oftalmología, otorrinolaringología, cirugía plástica y psiquiatría.

- **Otros procesos:**

  El servicio de emergencias desarrolla programas de mejora continua de la calidad, registros de estadísticas de morbilidad, mortalidad, así como, custodiar los archivos clínicos (dos (2) – tres (3) años).

- **Personal:**

  El servicio cuenta con un jefe de emergencias que sea médico especialista en emergencias y en el caso de un hospital con residencia en emergencias, con un director médico de emergencias, con disponibilidad veinticuatro (24) horas. Así mismo, posee una estructura de enfermeras de manera permanente, con una jefa de enfermeras, con el currículo y experiencia necesarias para la organización y operación del servicio, con capacitación en emergencias médicas y disponibilidad las veinticuatro (24) horas del día.

- **Rol del servicio:**

  Es el aspecto de mayor relevancia para el funcionamiento y gestión del servicio, debido a que se compromete a contribuir a mejorar la salud de la población atendiéndola de manera oportuna, con equidad, calidad, calidez, eficiencia y eficacia.

De este modo se puede decir que los servicios de emergencias son cuartos con equipos y médicos especializados que tratan con prioridad las enfermedades o accidentes más complicados como traumas, problemas cardíacos y quemaduras. En particular, el servicio de emergencia del Hospital Central Universitario " Dr. Antonio María Pineda" (HCUAMP), según el Informe de Movimiento Hospitalario (2000), es un hospital tipo IV (con 737 camas presupuestadas) que funciona como un centro asistencial de referencia regional que atiende a los estados Lara, Yaracuy, Falcón, Portuguesa y Barinas. Así mismo le presta servicio a otros estados principalmente Carabobo, Zulia y Trujillo.

Cabe destacar que un hospital Tipo IV se refiere a un hospital que se encuentra ubicado en poblaciones mayores de 100.000 habitantes y con un área de influencia de 1.000.000 habitantes, tienen más de 300 camas, cuentan con medicina interna, pediatría, cirugía general y ginecología – obstetricia, laboratorio, rayos X veinticuatro (24) horas, odontología, cirugía ambulatoria y especializada, sub-especialidades, unidades de larga estancia, terapia intensiva, anatomía patológica, docencia de pre y post grado. Dependen de la Corporación de Salud. De acuerdo a la red hospitalaria del portal de la gobernación del estado Aragua (www.aragua.gob.ve).

Según entrevista realizada al Coordinador del Postgrado de Emergencia del Decanato de Ciencias de Salud de la Universidad Centroccidental "Lisandro Alvarado" (UCLA), el servicio de emergencia, actualmente posee una distribución física y una estructura del equipo de profesionales, tal como se muestran en los cuadro 1 y 2 respectivamente.

Cuadro 1:

Estructura física del Servicio de Emergencia

Espacio Físico	Número de Camas
Sala de Observación	46 camas
Sala de Trauma y shock	6 camas
Sala de Cura	20 (divanes)

Fuente: Coordinación de Postgrado de Emergencia del HCUAMP (2008)

Cuadro 2:

Estructura del equipo profesional adscrito al Servicio de Emergencia del Hospital Central Universitario Antonio María Pineda

Equipo Profesional	Número de Profesionales
Jefe de Servicio	1
Especialista de Emergencia	3
Médicos Residentes(RI, RII, RII)	6 (dos de cada tipo)
Medico Residente de Medicina Interna	1
Estudiantes de Medicina Pregrado	10
Profesionales de Enfermería	47
Ayudantes	12

Fuente: Coordinación de Postgrado de Emergencia del HCUAMP (2008)

El proceso de atención de pacientes en el servicio de emergencia del HCUAMP, puede ser descrito de la siguiente forma: inicialmente los pacientes ingresan al servicio, son atendidos en la sala de curas por los estudiantes de medicina y por los médicos residentes (proceso de triage) que se encargan de examinarlos, diagnosticarlos y referirlos al sitio correspondiente de acuerdo al siguiente esquema:

(a) A la sala de observación, si amerita permanecer en observación o requiere ser hospitalizado; de requerir hospitalización es atendido en el servicio mientras se le da

la orden de traslado a la unidad que le corresponde (UCI, sala de quemados, traumatología, cardiología, ginecología o sala de maternidad, agudos, subagudos, cirugía, medicina de mujeres o medicina de hombres).(b) Al ambulatorio que le corresponde (según la ubicación de su residencia), si presenta una enfermedad que no debe ser tratada en el hospital, es decir, sino presenta una enfermedad de gravedad que son atendidas en este tipo de hospital, o al especialista si requiere de consulta con un especialista específico.

De igual manera los estudiantes de medicina y los médicos residentes luego de atender y referir al paciente, realizan un breve registro del mismo, en un cuaderno que se encuentra en la sala, en donde colocan los datos del paciente y una breve descripción del diagnóstico que presenta, es decir que no se cuenta con un registro detallado de la totalidad de pacientes atendidos, debido a que a los pacientes que se le crean historias médicas son solamente los que son referidos a la sala de observaciones. La búsqueda de la información se hace tediosa, debido a que hay que revisar línea por línea del cuaderno para obtener información necesaria, para la realización de análisis y estadísticas. (Por tal razón, es importante brindar un mecanismo de registro del paciente, que permita facilitar la información de los pacientes que acuden al servicio y así poder obtener información detallada para realizar los análisis y estadísticas, como por ejemplo saber cuáles son los pacientes que fueron atendidos en el servicio de emergencia y cuáles fueron referidos).

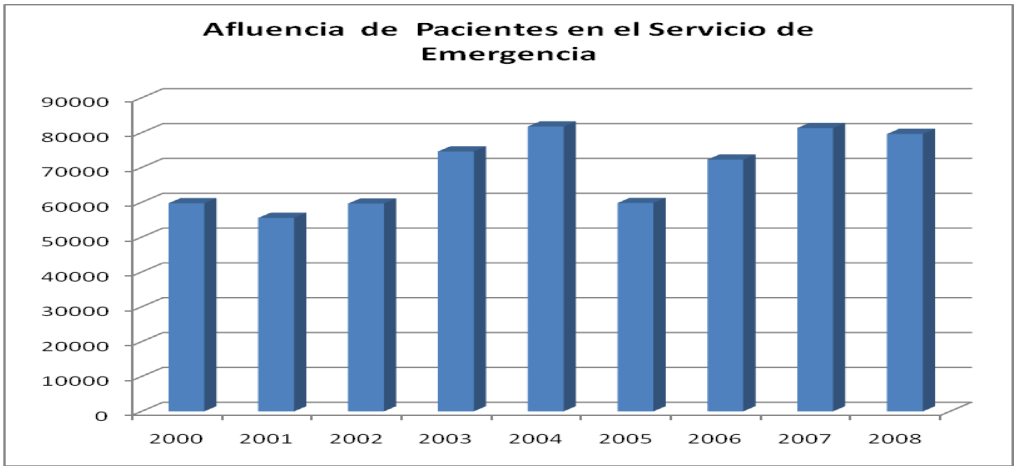
De acuerdo al levantamiento de información realizado en la unidad de Archivo de Historias Medicas, de la Coordinación de Estadística del HCUAMP, durante los últimos nueve (9) años, se observó una variabilidad en el número de pacientes que acuden al servicio de emergencia.

La experiencia en el área indica que los factores que inciden en la variabilidad de la afluencia de pacientes son: los días de cobro, periodos vacacionales, cambios de clima y cambios atmosféricos, epidemias de virus y eventos sociales, lo cual trae como consecuencia una saturación crónica al servicio.

En la gráfica de la siguiente página, se muestra la afluencia de pacientes en el servicio de emergencia del HCUAMP en el lapso comprendido en el año 2000 al 2008. Así mismo se puede observar la variabilidad de la afluencia de pacientes por año. Si consideramos la capacidad del servicio de emergencia descrita en los cuadros 1 y 2, y tomando como referencia el año 2001 en el cual se obtuvo la menor cantidad de pacientes atendidos, cincuenta y cinco mil quinientos cuarenta y ocho (55548) podemos calcular que el total de pacientes fue de cuatro mil seiscientos veintinueve (4629) mensual, ciento cincuenta y cuatro (154) pacientes diarios. Siendo más específicos nos da un resultado de seis (6) a siete(7) pacientes por hora, estimando examinar a cada paciente en 10 minutos, lo cual es humanamente difícil, debido a que el tiempo estimado en atender a un paciente es mayor al resultado obtenido por la gráfica.

Figura 1

Pacientes atendidos en el Servicio de Emergencia por año



Teóricamente, el tiempo estimado para la atención de los pacientes es de 15 minutos, tomando en cuenta que un paciente es atendido por un número que no puede ser determinado a priori por profesionales en el área, sin embargo, en la práctica este

tiempo no se cumple debido a que existen pacientes que requieren de atenciones especiales y sobrepasan el periodo de tiempo establecido.

Así mismo, el servicio de emergencia posee una actividad continua, es decir, presta sus servicios las veinticuatro (24) horas del día, todos los días del año, lo cual trae como consecuencia, la necesidad de contar con un equipo de profesionales que se encarguen de cubrir el servicio a lo largo del año. En el cuadro de la siguiente pagina, se muestra la distribución de los turnos del equipo de profesionales.

Cuadro 3:

Distribución de turnos del equipo profesional adscrito al Servicio de Emergencia del Hospital Central Universitario "Dr. Antonio María Pineda"

Tunos	Especialista en Emergencia	Médico Residente	Médico Residente Medicina Interna	Estudiante Pregrado Medicina	Profesional Enfermería	Profesional Enfermería Sala de Trauma y Shock	Ayudantes
Mañana	3	3	1	10	7	9	4
Tarde	3	3	1	10	7	9	4
Noche	1	3	1	0	7	8	4
Fines de Semana Feriados	1	3	1	0	7	8	4

Fuente: Coordinación de Postgrado de Emergencia del HCUAMP (2008)

Debe señalarse que el número de personas que solicitan el servicio simultáneamente es mayor a la capacidad del servicio, lo cual trae como consecuencia la necesidad de contar con un esquema de prioridad para la atención de los mismos. Este esquema de prioridad toma como consideración fundamentalmente dos factores, el primero la edad del paciente (de menor a mayor) y en segundo término el compromiso vital del paciente.

Si bien es cierto que, la afluencia es un factor que incide en la saturación del servicio, existen otros factores que influyen de forma determinante en el mismo, estos son: la capacidad del servicio, la demora en la exploración del paciente, la atención de pacientes con enfermedades que no ameritan ser atendidas en el servicio, pacientes en el servicio en espera por traslado a otras unidades del HCUAMP, entre otros.

Adicional a la situación anteriormente planteada, es importante acotar la problemática de carácter social que presentan los familiares que acompañan al paciente, debido al estrés generado por la falta de información respecto al estado del paciente, por la inexistencia de un mecanismo que proporcione información adecuada y oportuna a dichos familiares.

Por todo lo anteriormente expuesto, es evidente la necesidad de ofrecer alternativas de apoyo que permitan coadyuvar el proceso asistencial y la ubicación definitiva del paciente con miras a mejorar el desempeño del servicio y minimizar el deterioro en la asistencia del paciente, así como también, el agotamiento y desmotivación del equipo de profesionales.

Para dar respuesta a la problemática anteriormente planteada, se propone el diseño del Sistemas Multiagente para la gestión de servicios de emergencia médica hospitalaria usando metodología INGENIAS, donde se conjugan técnicas inteligentes con respuestas en tiempo real, en un entorno distribuido, utilizando Agentes, para emular los procesos de la gestión de servicio de emergencia; por su capacidad de manejar grandes volúmenes de datos y poseer tolerancia a fallos, gracias a su capacidad de cooperación que permite que un agente que presente fallos durante el proceso, pueda ser sustituido por otro sin necesidad de retomar los pasos desde el principio o perder información.

Objetivos de la Investigación

Objetivo General

Diseñar un Sistema Multiagente para la Gestión de Servicios de Emergencia Medica Hospitalaria usando Metodología INGENIAS.

Objetivos Específicos

1. Diagnosticar la situación actual del servicio de Emergencia del Hospital Central "Dr. Antonio María Pineda", en cuanto a la gestión de Servicio de Emergencia Hospitalaria.

2. Determinar la factibilidad de diseñar un Sistema Multiagente para la gestión de servicios de la emergencia del Hospital Central "Dr. Antonio María Pineda".

3. Modelar la arquitectura del Sistema Multiagente para la gestión de servicios de la emergencia del Hospital Central "Dr. Antonio María Pineda" usando Metodología INGENIAS.

Justificación e Importancia

La razón básica que justifica la presente investigación, es la necesidad que tiene el Hospital Central Universitario "Dr. Antonio María Pineda" de mejorar la gestión del servicio de emergencia a objeto de encontrar una propuesta que sirva como apoyo para que dicha gestión se realice de la manera más efectiva y que satisfaga los requerimientos, tanto del personal que labora como del paciente que lo solicita.

El estudio permitirá obtener información descriptiva del proceso de gestión de servicio, con el fin de que los pacientes y sus familiares obtengan una mejor atención. En este sentido, la presente investigación es importante porque contribuye a dar información veraz y confiable a los médicos, que son parte fundamental en el proceso y que finalmente se asegure la integración y el mantenimiento de relaciones adecuadas entre los gestionadores del proceso de servicio, pacientes y familiares, puesto que estos serán beneficiados porque contaran con información del estado de los pacientes.

La investigación introduce una nueva perspectiva de análisis y aporta al conocimiento sobre el diseño de un Sistema Multiagente usando Metodología INGENIAS para la gestión de servicio de emergencia médica hospitalaria, para lograr la mayor efectividad y eficiencia posible en el proceso.

Los resultados de la investigación permitirán ofrecer un mejor servicio tanto a los pacientes, familiares y a al personal que labora en la emergencia debido a que contarán con información importante sobre sus requerimientos, de esta forma se logrará mejorar el tiempo de atención y respuesta, debido a que los pacientes serán ubicados en los sitios que le corresponden, es decir, que si presentan un caso que se considere de emergencia serán atendidos, en caso contrario serán referidos al lugar donde les correspondan, informando a los familiares el lugar donde fue referido el paciente.

Alcances y Limitaciones

a) El sistema será diseñado basándose en el funcionamiento actual del servicio de emergencia del HCUAMP, de Barquisimeto Edo Lara Venezuela.

b) Para el diseño del sistema se modelaran los diferentes agentes incluyendo en ellos el diseño de las tablas de datos, entradas (percepciones) salidas (acciones) y funcionamiento.

c) Se desarrollara un prototipo para demostrar las interfaces de usuarios y el funcionamiento del sistema como aporte para su futuro desarrollo completo.

d) Este sistema servirá de modelo para otros servicios de emergencia con características similares.

e) Este trabajo servirá de aporte para futuras investigaciones que trabajan sobre mejoramiento de procesos de gestión, sistemas multiagente y metodología INGENIAS.

f) El sistema proveerá información a los pacientes, familiares y médicos que allí laboran a través de un agente de consulta el cual mostrará el estado del paciente.

g) Se podrá llevar el registro de todos los pacientes que asisten a la sala, por medio de un agente de registro de pacientes, así como también los que fueron referidos a las consultas de especialidad o al ambulatorio, teniendo como beneficio llevar de manera automatizada el control de los mismos.

h) Se tendrá el registro de los médicos que laboran en la sala con sus respectivos turnos, a través del agente de registrar médicos.

i) El diseño del sistema contará con una arquitectura de agentes distribuida.

j) Para el diseño del Sistema se utilizará la metodología de agentes INGENIAS.

CAPITULO II

MARCO TEÓRICO

Antecedentes

Según Claret (2005), "Los antecedentes se refieren a la revisión de trabajos previos sobre el tema en estudios realizados fundamentalmente en instituciones de educación superior reconocidas o, en su defecto en otras organizaciones". Por tal razón para la realización de esta investigación se estudiaron distintos trabajos relacionados con el área, de los cuales los más resaltantes fueron:

García (2006), entre otros, presentan el trabajo **"Modelando el Proceso de Desarrollo de INGENIAS con EMF"** realizado en el departamento de Ingeniería de Software e Inteligencia Artificial de la Universidad de Complutense España. En este trabajo se muestra de forma práctica cómo utilizando un estándar de modelado se obtiene la definición del proceso de desarrollo asociado a una metodología orientada a agentes concreta. A partir de dicha especificación se esboza el modo de construir una herramienta de soporte al modelado de procesos usando el estándar de Object Management Group (OMG), SPEM (Software Process Engineering Metamodel) y herramientas, como Eclipse Modeling Framework (EMF), que facilitan la construcción del software necesario.

El proceso anterior proporciona múltiples posibilidades, ya que simplifica enormemente la construcción de herramientas software de soporte a la definición de procesos. Además, el hecho de modelar el proceso de desarrollo de una metodología es fundamental para facilitar el aprendizaje de la misma y su utilización diaria.

Los resultados obtenidos permiten guiar al desarrollador de un SMA en los pasos a dar en su construcción, desde la definición de requerimientos hasta la implementación. De modo que una herramienta de soporte metodológico, que contemple el proceso definido, podrá guiar al desarrollador paso a paso, indicándole lo que debe hacer a continuación o comprobando, al realizar una definición, que se han cumplimentado los pasos previos que son necesarios. Dicho trabajo provee a la investigación información acerca de la forma práctica de realizar un diseño de un sistema multiagente utilizando metodología INGENIAS.

Serrano y Suros (2005), presentaron un "**Diseño de un Sistema Multiagente para la Gestión de Recursos usando la Metodología MAS CommonKADS"** realizado en la Facultad de Ciencias de la Universidad Central de Venezuela. En este trabajo se presenta el proceso de modelado de un sistema para la gestión de recursos en redes de alta velocidad. Para el diseño se utiliza la metodología orientada a agentes MAS CommonKADS que cubre el desarrollo de todo el ciclo de vida de un sistema multiagente a través del desarrollo de siete modelos, sin presentar ningún tipo de restricción con respecto al número de agentes presentes en él. Su uso se basa en procedimientos bien conocidos que facilitan el aprendizaje de la metodología y además, define plantillas textuales para cada modelo que permiten la descripción de los agentes, sus características y la estructura del Sistema Multiagente (SMA).

Como resultado se obtienen los modelos de agentes, tareas, comunicación, coordinación, organización, experiencia y diseño. Para cada modelo de la metodología, se demostró el proceso estándar de desarrollo y la notación gráfica. El esfuerzo se centró en la implementación de la metodología en el modelado de un sistema basado en agentes distribuidos. Las redes de alto rendimiento son redes de mucha complejidad y el paradigma orientado a agentes contribuye significativamente en el diseño de soluciones adecuadas.

En este sentido, la utilización de MAS CommonKADS en el modelado de un SMA como parte de la Inteligencia Artificial Distribuida, demuestra que se pueden obtener

soluciones para un problema de alta complejidad, permitiendo el desarrollo de todo el ciclo de vida del software de manera general. MAS CommonKADS es lo suficientemente completa para permitir el desarrollo de sistemas multiagentes robustos y complejos.

Sin embargo, a pesar de que la metodología CommonKADS tiene un modelo de agente, no cubre aspectos relevantes de los agentes, es decir, no ofrece facilidad alguna para modelar interacciones entre agentes, lo que dificulta su aplicabilidad en el caso de SMA. Una extensión interesante de esta investigación es el uso de metodologías de diseño para sistemas multiagentes que integren un ambiente que permita generar códigos. El trabajo descrito anteriormente aporta a la investigación, el estudio de diseñar un sistema para la gestión de servicio.

Gómez (2003), presenta un trabajo sobre **"Metodologías para el desarrollo de sistemas multi-agente",** realizada en el departamento de sistemas informáticos y programación de la facultad de Informática de la Universidad de Complutense España. Las metodologías estudiadas son representantes de diferentes líneas de investigación en agentes: agentes como sistemas basados en concimiento (MAS-CommonKADS), procesos concurrentes (MaSE), agentes BDI (BDI, ZEUS), agentes según la definición de Newell (INGENIAS) o agentes como entidades computacionales que integran diferentes tecnologías (Vowel Engineering).

La aplicación de estas metodologías depende mucho de la formación que tenga su usuario final. Esto quiere decir que son interpretadas en función de los intereses de quien las aplica. De momento, no hay soluciones que fuercen un uso concreto. El trabajo descrito anteriormente sirvió de base fundamental para analizar las distintas metodologías orientadas a agentes y de esta manera poder seleccionar la que se va a utilizar para este trabajo la cual es **INGENIAS**.

Salvador (1996), en el trabajo titulado: **"El Sistema de Información de Conjuntos Minimos de Datos Basicos (CMBD) como herramienta de gestión y de control de calidad asistencial",** realizado en el Departamento de Microbiología,

Medicina Preventiva y Salud Pública, de la Facultad de Medicina de la Universidad de Zaragoza, España.

Este trabajo pretende mejorar la calidad de la información proporcionada habitualmente por el CMBD y ampliarla con indicadores de calidad asistencial. Los objetivos son demostrar que la estancia media no es un indicador válido, y se proponen los m-estimadores como alternativa; y se demuestra que la estructura del CMBD es totalmente inadecuada para reflejar con precisión y exactitud la actividad asistencial y producción de cada servicio hospitalario.

Se concluye que para calcular el coste por proceso, es necesario que estén codificadas el 100% de las altas hospitalarias, estando a cargo la codificación de personal médico, siendo la fuente de datos el informe de alta si esta correctamente cumplimentado o bien la historia clínica. La calidad de los datos es fundamental en el éxito de cualquier sistema de información, siendo el médico el elemento clave ya que es quien debe de especificar todas las patologías y procedimientos realizados al paciente con precisión y exhaustividad, influyendo todo esto en la asignación correcta del paciente, y en el cálculo del coste por proceso.

Bases Teóricas

Inteligencia Artificial:

De acuerdo con Russell y Norvig (1996). Se considera que la inteligencia artificial (IA) es el campo de investigación que se enfoca a lograr la construcción y comprensión de las entidades inteligentes. Así se puede decir que la IA, es una disciplina relativamente nueva que presenta su inicio en 1956, y que tiene diversas definiciones. Dichos autores sustentan que de este concepto se puede diferenciar dos grandes grupos: los que definen la inteligencia artificial relacionándose a procesos mentales y al razonamiento, y, de otro lado, las que se enfocan aplicándose en la conducta.

En el siguiente cuadro se pueden observar algunas definiciones de IA de acuerdo a diferentes autores citados por Russell y Norving (1996). Las definiciones varían en torno a dos dimensiones principales. Las que aparecen en la parte superior se refieren a procesos mentales y al razonamiento, en tanto que la parte inferior aluden a la conducta. Por otra parte, las definiciones de la izquierda miden la condición deseable en función de eficiencia humana, mientras que las de la derecha lo hacen en conformidad con un concepto de inteligencia ideal, denominado racionalidad.

Cuadro 4:

Definiciones de Inteligencia Artificial (IA)

"La interesante tarea de lograr que las computadoras piensen… máquinas con mente, en su amplio sentido literal." (Haugeland, 1985.) "[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades tales como toma de decisiones, resolución de problemas, aprendizaje…" (Bellman, 1978.)	"El resultado de las facultades mentales mediante el uso de modelos computacionales." (Charniak y McDermott, 1985.) "El estudio de los cálculos que permiten percibir, razonar y actuar." (Winston, 1992.)
"El arte de crear máquinas con capacidad de realizar funciones que realizadas por personas requieren de inteligencia." (Kurzweil, 1990.) "El estudio de cómo lograr que las computadoras realicen tareas que, por el momento, los humanos hacen mejor." (Rich y Knight, 1991.)	"Un campo de estudio que se enfoca a la explicación y emulación de la conducta inteligente en función de procesos computacionales." (Schalkoff, 1990.) "La rama de la ciencia de la computación que se ocupa de la automatización de la conducta inteligente." (Luger y Stubblefield, 1993.)
Algunas definiciones de IA. Se agrupan en cuatro categorías	
Sistemas que piensan como humanos	Sistemas que piensan racionalmente
Sistemas que actúan como humanos	Sistemas que actúan racionalmente

Fuente: Russell y Norvig (1996).

Según autores mencionados, para definir un sistema que piense como humano, hay que imitar la forma en como los seres humanos construyen el pensamiento, así como la forma en que llegan a una conclusión y el tiempo que se demoran

para hacerlo; por ello en la ciencia cognoscitiva hay modelos computacionales de IA y técnicas experimentales de psicología, para intentar elaborar teorías precisas y verificables del funcionamiento de la mente humana.

De otro lado, para definir un sistema que piense racionalmente, se lleva a cabo el enfoque de las leyes del pensamiento, en el cual se llega a fundamentos de la lógica mostrando que la mente siempre tiene una "manera correcta de pensar", si se llega a conclusiones correctas partiendo de premisas correctas, esto se puede lograr a través de silogismos. Este enfoque de leyes, lleva a generar una tradición logicista en IA, que permite crear sistemas inteligentes, que describan problemas en notación lógica y así encontrarle solución, siempre y cuando ésta exista.

Inteligencia Artificial Distribuida:

De acuerdo con la Dirección Nacional de Servicios Académicos Virtuales de la Universidad Nacional de Colombia. La inteligencia artificial distribuida (IAD) aparece en la década de los 80's como una nueva rama de la inteligencia artificial (IA) que tiene el fin de estudiar sistemas inteligentes formados por un conjunto de varios agentes, estos intentan resolver problemas en donde una conducta colectiva es más eficiente que una conducta individual, como lo estudia la inteligencia artificial que hace el análisis de un único agente que se encuentra en un ambiente no cambiante y que intenta resolver todo el problema con solo esta entidad.

Siguiendo con el mismo orden de ideas, la dimensión y la complejidad de los nuevos sistemas de información son cada vez mayores, los planes para encontrar una solución global ante cierto problema necesitan integrar soluciones de problemas más pequeños. Lo anterior se asemeja a la idea de "divide y vencerás", en la cual los planes para resolver subproblemas son más simples y precisos.

Razones de la transición de la IA a la IAD:

La primera razón técnica es que en esta época muchos problemas son esencialmente distribuidos y la segunda es la integración de los sistemas de IA para mejorar la capacidad mediante la distribución del conocimiento lo que conlleva a un manejo descentralizado ofreciendo las siguientes ventajas:

- Incremento de la flexibilidad: Se permite la adición de nuevos agentes.
- Mejor seguridad y efectividad: Los agentes se pueden especializar en una tarea específica.
- Mejor tiempo de respuesta: Los agentes pueden resolver sus problemas particulares al mismo tiempo.
- Reducción de la complejidad: Una tarea puede ser descompuesta en varias subtareas y asignarlas a los agentes.
- Reutilización: La solución presentada por un agente en un sistema puede ser incorporada a otro.

La Inteligencia Artificial Distribuida es una subrama de la IA que se centra en la resolución de problemas mediante aplicación tanto de técnicas de la Inteligencia Artificial como de múltiples solucionadores de sistemas. Se involucran además mínimo dos agentes que funcionarían como solucionadores de problemas, estos agentes serían autónomos o semi-autónomos, tendrían un cierto conocimiento del problema y serían razonables.

Problemas básicos de la Inteligencia Artificial Distribuida:

Los problemas básicos que estudia la IAD, de acuerdo con la Dirección Nacional de Servicios Académicos Virtuales de la Universidad Nacional de Colombia (2009), que serán comunes a todos los sistemas son:

- Cómo formular, describir, descomponer y asignar problemas y cómo sintetizar los resultados en un grupo de agentes inteligentes.

- Cómo capacitar a los agentes para que se comuniquen e interactúen, es decir, qué tipo de lenguaje y qué protocolos de comunicación utilizarán, qué y cuándo se comunicarán, entre otros.

- Cómo asegurar que los agentes actúen coherentemente a la hora de tomar decisiones o de ejecutar una acción, equilibrando los efectos globales de las decisiones locales y evitando interacciones perjudiciales.

- Cómo diseñar los agentes individuales de manera que puedan representarse y razonar acerca de las acciones, los planes y el conocimiento de otros agentes, de modo que puedan coordinarse.

- Cómo reconocer y reconciliar puntos de vista diferentes e intenciones conflictivas.

- Cómo utilizar técnicas de ingeniería de software y desarrollar sistemas con IAD.

Áreas de trabajo de la IAD:

Según la Dirección Nacional de Servicios Académicos Virtuales de la Universidad Nacional de Colombia (2009). Los anteriores problemas básicos que intenta atacar la IAD se solucionan mediante diferentes teorías que se ven reflejadas en distintas áreas de trabajo, las cuales se pueden descomponer en cuatro perspectivas:

- **Perspectiva de grupo:** Estudia las teorías y técnicas que caracterizan a un grupo de agentes, es decir los métodos necesarios para formar una sociedad de agentes para que exista cierto grado de planeación, coordinación, comunicación y coherencia entre sus entidades.

- **Perspectiva de agente:** Estudia la entidad agente, arquitecturas para desarrollo de agentes, lenguajes para desarrollo y comunicación de agentes,

clasificación de tipo comercial y estructural, además como puede este cooperar en la sociedad de agentes.

- **Perspectivas Particulares:** Estudia las relaciones existentes de campos de la informática como la IA, Ingeniería de Software con la IAD. Por ejemplo sistemas de información abiertos que reaccionan a casos imprevistos, por lo que son inconsistentes, asíncronos, concurrentes, con control descentralizado; ecosistemas para evaluar un agente o la sociedad de agentes en base a una analogía ecológica y ver como esta evoluciona con el paso de las iteraciones; ingeniería de software basada en agentes los agentes encapsulan los programas y mediante la definición de unas primitivas, permiten el intercambio de órdenes y datos entre los programas.

- **Perspectivas del diseñador:** Estudia metodologías y herramientas para poder desarrollar software basado en agentes. Las metodologías asisten al desarrollador de sistemas de IAD en el ciclo de vida, y las herramientas le sirven en la implementación de dichos sistemas.

Tipos de sistemas de la IAD:

De acuerdo con la Dirección Nacional de Servicios Académicos Virtuales de la Universidad Nacional de Colombia, los tipos de sistemas de la IAD son:

- **Solución de Problemas Distribuidos (DPS):** Se considera como el trabajo de solucionar un problema en particular, este problema se divide entre los nodos que conforman el conjunto de agentes, estos cooperan y comparten conocimiento sobre el problema y la solución desarrollada. En el sistema DPS todos los agentes tienen conocimiento homogéneo y completo del problema así como de la solución, están interconectados y el desarrollo de la solución se obtiene a partir de una secuencia de operaciones.

- **Sistemas MultiAgente:** Los Sistemas MultiAgente (MA) se preocupan por coordinar la conducta inteligente de agentes autónomos. Estos agentes hacen parte de una colección y pueden coordinar su conocimiento, objetivos, habilidades y planes juntamente para tomar una acción o resolver una meta global, al igual que los DPS estos pueden tener conocimiento parcial sobre el problema y las soluciones. En estos sistemas debe haber un proceso de racionalización para la coordinación del conjunto de agentes.

  Por lo general en estos sistemas MA los agentes con sus creencias, deseos e intenciones construyen el problema y el plan o secuencia de acciones para solucionarlo.

Agentes:

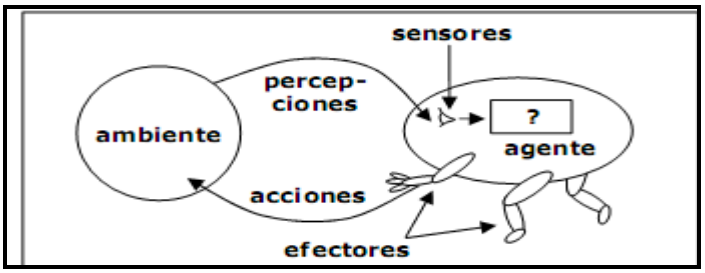
Según Russell y Norvig. (1996), definen un agente como todo aquello que pueda considerarse que percibe su ambiente mediante sensores y que responde o actúa en tal ambiente por medio de efectores. Los agentes humanos tienen ojos, oídos y otros órganos que le sirven de sensores, así como manos, piernas, boca y otras partes de su cuerpo que le sirven de efectores. En el caso de los agentes robóticos, los sensores son sustituidos por cámaras y telémetros infrarrojos, y los efectores son reemplazados mediante motores. En el caso de un agente de software sus percepciones y acciones vienen a ser las cadenas de bits codificados.

Por otra parte, Wooldridg (1995) lo define como cualquier proceso computacional dirigido por el objeto capaz de interactuar con su entorno de forma flexible y robusta.

En la figura de la siguiente pagina, se muestra la definición de agentes en forma gráfica de acuerdo con Russell y Norving (1996).

Figura 2

Los Agentes interactúan con los ambientes a través de sensores y efectores



Fuente: Russell y Norvig. (1996)

Características de los Agentes:

Según Wooldridge (1995) los agentes presentan las siguientes características:

- **Autonomía:** Capacidad de actuar sin intervención humana directa o de otros agentes.
- **Sociabilidad:** Capacidad de interaccionar con otros agentes, utilizando como medio algún lenguaje de comunicación entre agentes.
- **Reactividad:** Un agente está inmerso en un determinado entorno (hábitat), del que percibe estímulos y ante los que debe reaccionar en un tiempo preestablecido.
- **Iniciativa:** Un agente no sólo debe reaccionar a los cambios que se produzcan en su entorno, sino que tiene que tener un carácter emprendedor y tomar la iniciativa para actuar guiado por los objetivos que debe de satisfacer.
- **Movilidad:** Habilidad de trasladarse en una red de comunicación informática.

- **Veracidad:** No comunica información falsa intencionadamente.

- **Benevolencia:** No tiene objetivos contradictorios y siempre intenta realizar la tarea que se le solicita.
- **Racionalidad:** Tiene unos objetivos específicos y siempre intenta llevarlos a cabo.

Arquitectura de los Agentes:

De acuerdo con Bravo (2003). Existen diferentes arquitecturas para diseñar y desarrollar agentes las cuales son mencionadas a continuación:

- **Arquitectura Basada en Lógica:** Este tipo de arquitectura surgió como una extensión directa de las investigaciones que en la lógica computacional se han realizado en los últimos años. Sugiere que la calidad inteligente de los agentes pueden ser construida usando un lenguaje basado en lógica. Estos lenguajes permiten representaciones simbólicas de hechos reales, y su manipulación sintáctica es hecha utilizando lógica de primer orden. Bajo este tipo de arquitectura cada agente procede a ejecutar acciones después de realizar un proceso deductivo, basado en una Base de Datos que hace el papel de las creencias humanas.

- **Arquitectura reactiva:** Se basan en una visión orientada a la percepción - control – acción. Los agentes en su definición deben ser capaces de percibir estados del ambiente, y actuar en función de ellos. Este tipo de arquitectura no considera una base de datos sin experiencia, creencias o hechos; solo se realiza la acción observando el estado del ambiente para proceder a modificarlo con la finalidad de llevar el entorno a su estado controlado.

- **Arquitectura Creencia – Deseo – Intención:** Tiende su funcionamiento en el razonamiento práctico humano, que actúa cada vez que producimos una decisión. Esto involucra los objetivos que se desean conseguir, las opciones que están disponibles, el conocimiento sobre el entorno, y las acciones que se deben ejecutar para lograr los objetivos propuestos. Podemos describir los componentes de esta arquitectura de la siguiente forma:

  Creencias: Es la base de datos de hechos y reglas que cada agente contiene. Es la base de conocimiento sobre el entorno en que se desenvuelve el agente. Las creencias en conjunto con la entrada del ambiente son procesadas con la finalidad de tomar decisiones.

  Deseos: Se definen como metas finales a alcanzar. En función de ellos se generan las opciones, alternativas o comportamientos a seguir para alcanzar objetivos propuestos. Utilizando la base de creencias se escoge la opción que se considera más acertada.

  Intenciones: Es el conjunto de acciones sin ejecutar como producto del sistema de creencias y deseos del agente.

- **Arquitectura por Niveles:** Esta arquitectura es la combinación de otras arquitecturas propuestas. En la mayoría de los casos los agentes deben poseer un conjunto de cualidades que no pertenecen a una única arquitectura. Por ejemplo deben ser reactivos en algunas veces, y en otros autónomos y pro-activos, o también, pueden ser un modelo basado en lógica que esté integrado con una arquitectura CDI. Es por esta razón que varios autores han propuesto una arquitectura que integre las características de cada una de las otras arquitecturas y que se distribuyan en forma de niveles. Los niveles pueden estar acoplados en dos posiciones: vertical, donde se obtiene una única salida, y la entrada es procesada por varios niveles de forma secuencial, u horizontal,

donde la entrada es procesada en forma paralela por los diferentes niveles y se obtienen diferentes propuestas de acciones.

Sistemas Multiagentes:

En este caso los agentes no son desarrollados de forma independiente sino como entidades que constituyen un sistema y que se relacionan con objetos, para ello es fundamental que interactúen por medio del diálogo, que se deleguen funciones, que cooperen para lograr un objetivo común y que coordinen lo que están haciendo, para organizar el proceso de la solución del problema y se eviten interacciones nocivas, de esta forma diversos tipos de agentes proporcionan servicios inteligentes a los usuarios. Los sistemas multiagente se abren paso en el campo de la Inteligencia Artificial Distribuida (DAI, Distributed Artificial Intelligence), ya que esta rama de la IA se encarga de su comprensión.

Weiss (1998), define los Sistema Multiagentes Sistema formado por un conjunto de componentes (semi)autónomos: Cada agente no tiene información completa ni la capacidad para resolver el problema. Tienen puntos de vista limitados. No hay un sistema de control global. Los datos están descentralizados (computación asíncrona).

Por otra parte Bravo (2003), sostiene que los Sistemas Multiagentes (SMAs), son sistemas de software que agrupan un conjunto de cualidades pertenecientes a los tópicos mas avanzados que en el área de computación se tratan en la actualidad. Estos tópicos tienen que ver con el uso de algoritmos inteligentes, sistemas distribuidos, programación lógica, ontologías para conversaciones, hilos de ejecución, maquinas virtuales, metalenguajes, interoperabilidad e integración de sistemas entre otros.

Así mismo Bravo (2003), afirma que esta conjunción de conceptos y tecnología, hace posible dar respuestas a problemas complejos, a través de la construcción de

sistemas autoregulados y dinámicos, que deben integrarse con aplicaciones legadas, plataformas y sistemas heterogéneos.

Metodologías para Sistemas Multiagentes:

Según Cuesta (2005), las Metodologías Orientadas a Agentes se han convertido en un importante campo de investigación. En los últimos años se han propuesto numerosas metodologías para facilitar y soportar el desarrollo de sistemas distribuidos complejos. Por tanto, cuando se plantea la construcción de una aplicación de agentes es importante decidir qué metodología utilizar.

En este sentido, las metodologías actuales definen elementos que les sirven para agrupar las distintas funcionalidades asociadas a un agente o a un grupo de agentes. Así aparecen conceptos como rol o servicio. Y por otro lado, para facilitar la comprensión de sistemas complejos tal y como ocurre con los lenguajes convencionales de implementación. "INGENIAS" (2005).

A continuación se presentara la descripción de algunas de las metodologías orientadas a agentes, según "INGENIAS" (2005):

MAS-CommonKADS:

Esta metodología extiende CommonKADS Tansley y Hayball (1993) aplicando ideas de metodologías orientadas a objetos y metodologías de diseño de protocolos para su aplicación a la producción de SMAs Iglesias (1998). La metodología CommonKADS gira en torno del modelo de experiencia (explicado más tarde) y está pensada para desarrollar sistemas expertos que interactúan con el usuario. De hecho considera sólo dos agentes básicos: el usuario y el sistema. Este hecho influye en el modelo de comunicación que, consecuentemente, trata de interacciones hombre-máquina.

Esta metodología ha sido la primera en hacer un planteamiento de SMA integrado con un ciclo de vida de software, concretamente el espiral dirigido por riesgos Pressman (1982). Propone siete modelos para la definición del sistema: agente, tareas, experiencia, coordinación, comunicación, organización y diseño. Cada modelo presenta una reseña a la teoría sobre la que se basa. El modelo en sí parte de una descripción gráfica que luego se complementa con explicaciones en lenguaje natural de cada elemento. Existe por cada modelo una descripción de las dependencias respecto de otros modelos y de las actividades involucradas. Estos modelos se hayan descritos ampliamente en Iglesias (1998) en lenguaje natural, complementándose con otras notaciones como SDL (Specification and Description Language) o MSC (Message Sequence Chart) para describir el comportamiento de los agentes cuando interaccionan.

Creencias, Deseos e Itenciones (BDI):

Las arquitecturas BDI se inspiran en un modelo cognitivo del ser humano Bratman (1987). Los agentes utilizan un modelo del mundo, una representación de cómo se les muestra el entorno. El agente recibe estímulos a través de sensores ubicados en el mundo. Estos estímulos modifican el modelo del mundo que tiene el agente (representado por un conjunto de creencias). Para guiar sus acciones, el agente tiene *Deseos*. Un *deseo* es un estado que el agente quiere alcanzar a través de *intenciones*. Éstas son acciones especiales que pueden abortarse debido a cambios en el modelo del mundo. Aunque la formulación inicial es de Bratman, fueron Kinny y Georgeff, (1997), quienes formalizaron este modelo y le dieron visos de metodología, de hecho es su trabajo lo que aquí se comenta.

Para especificar el sistema de agentes, se emplean un conjunto de modelos que operan a dos niveles de abstracción: externo e interno. Primero, desde un punto de vista externo, un sistema se modela como una jerarquía de herencia de clases de

agentes, de la que los agentes individuales son instancias. Las clases de agente se caracterizan por su propósito, sus responsabilidades, los servicios que ejecutan, la información acerca del mundo que necesitan y las interacciones externas. Segundo, desde un punto de vista interno, se emplean un conjunto de modelos (modelos internos) que permiten imponer una estructura sobre el estado de información y motivación de los agentes y las estructuras de control que determinan su comportamiento (creencias, objetivos y planes en este caso).

En esta metodología, la integración con el ciclo de vida de software es muy reducida. Los autores proponen una serie muy reducida de pasos para generar los modelos. Estos pasos se repiten haciendo que los modelos, que capturan los resultados del análisis, sean progresivamente elaborados, revisados y refinados. Además, el refinamiento de los modelos internos conlleva la realimentación de los modelos externos. Por ejemplo, el construir los planes y creencias de una clase de agente clarifica qué nivel de detalle se requiere en la representación del mundo que posee el agente.

Además, sobre la arquitectura que soporte estos modelos se imponen varias restricciones: que asegure que los eventos se responden en su momento, que las creencias se mantengan consistentemente, y que la selección de planes y ejecución se desarrolle de manera que refleje ciertas nociones de racionalidad.

El modelo BDI ha sido importante para este trabajo, como aplicación del principio de racionalidad. Proporciona una definición, basada en elementos intuitivos, de cómo actúan los agentes. Esta definición es lo suficientemente genérica como para permitir múltiples implementaciones sobre diferentes plataformas.

Metodológicamente, la propuesta de Kinny y Georgeff (1997), es consecuente con la dificultad de generar los modelos que proponen. Como ellos admiten, existen dependencias entre los diferentes modelos que dificultan el proceso de generación de

los modelos que describen el SMA. Como solución proponen un proceso iterativo e incremental (como el Proceso Racional Unificado) con realimentaciones. Sin embargo, la forma en que tienen lugar estas realimentaciones no queda completamente definida.

Multi-agent systems Software Engineering (MASE):

DeLoach (2001), se concibe como una abstracción del paradigma orientado a objetos donde los agentes son especializaciones de objetos. En lugar de simples objetos, con métodos que pueden invocarse desde otros objetos, los agentes se coordinan unos con otros vía conversaciones y actúan proactivamente para alcanzar metas individuales y del sistema.

En MaSE los agentes son simplemente una abstracción conveniente, que puede o no poseer inteligencia. En este sentido, los componentes inteligentes y no inteligentes se gestionan igualmente dentro del mismo armazón. Dado el enfoque inicial, los agentes se ven como especializaciones de objetos. De hecho, el sistema se construye sobre técnología orientada a objetos y su aplicación a la especificación y diseño de sistemas multi-agente.

El análisis en MaSE consta de tres pasos: capturar los objetivos, aplicar los casos de uso y refinar roles. El diseño consta de cuatro pasos: crear clases de agentes, construir conversaciones, ensamblar clases de agentes y diseño del sistema. La mayoría de estos pasos se ejecutan dentro de la herramienta que soporta MaSE, AgentTool DeLoach (2001). Como productos de estas etapas, MaSE espera: diagramas de secuencia para especificar interacciones, diagramas de estados para representar procesos internos a las tareas y modelar interacciones, descomposición del sistema (agente) en subsistemas (componentes del agente) e interconexión de los mismos (definición de la arquitectura del agente). Estos elementos son característicos del UML (Lenguaje Unificado de Modelado), de hecho su uso recuerda mucho a

SDL. Está por ver si las especificaciones resultantes, son capaces de expresar elementos característicos como razonamiento de los agentes, organización de los agentes o caracterización de su estado mental.

Además, la integración de estos diagramas en el proceso de desarrollo parece demasiado simple. Al final, la metodología podría traducirse como tome la herramienta de soporte y rellene los diferentes apartados. Esto supone ignorar que, como en el modelo BDI, se tienen dependencias entre los diagramas propuestos (como entre los diagramas de secuencia y las conversaciones) y que no es tan sencillo el saber qué máquinas de estados definen la ejecución de una tarea en el contexto de una interacción.

La herramienta de soporte permite generar código automáticamente a partir de la especificación. La generación de código es independiente del lenguaje de programación utilizado, ya que se realiza recorriendo las estructuras de datos generadas en la especificación y generando texto como salida. No obstante, el proceso de generación de código es mejorable, ya que el código de los componentes a generar está entremezclado con el código que lee la especificación.

ZEUS:

ZEUS consta de una herramienta y una metodología, Collis y Ndumu (1999), de forma similar a AgentTool y MaSE. Desde su aparición, ZEUS se ha convertido en referencia de cómo debe ser una herramienta para el desarrollo de SMA. Sobre todo, por la forma en que combinan los distintos resultados de investigación en agentes (planificación, ontologías, asignación de responsabilidades, relaciones sociales entre agentes) en un sistema completamente funcional. De hecho, la aplicación genera incluso programas para arrancar el sistema especificado e incluye herramientas de monitorización como el Visor de Sociedad que muestra los agentes existentes y sus relaciones, la Herramienta de Control para ver o modificar remotamente el estado de

los agentes y los Generadores de Informes para obtener estadísticas de funcionamiento e informes de actuación de la sociedad de agentes.

La metodología ZEUS propone un desarrollo en cuatro etapas, Collis y Ndumu (1999), el análisis del dominio, el diseño de los agentes, la realización de los agentes y el soporte en tiempo de ejecución. Las etapas soportadas por la herramienta son la de realización de los agentes y la de soporte en tiempo de ejecución. Las etapas anteriores se basan en el uso de roles para analizar el dominio y en su asignación a agentes.

Ante la similitud de enfoques, se impone una breve comparativa entre ZEUS y MaSE. Conceptualmente, ZEUS es superior a MaSE. Si bien la primera está más orientada a la aplicación de tecnología de agentes (planificación, definición de ontologías, secuenciación de tareas), la segunda se orienta más a las prácticas de ingeniería convencional. Metodológicamente, ZEUS es más pobre que MaSE. El modelado de roles, propuesto en Collis y Ndumu (1999), no profundiza en la aplicación de la herramienta dentro del proceso de desarrollo. El ámbito de la metodología se limita a estudiar cómo agrupar la funcionalidad del sistema dentro de cada rol, dejando aparte consideraciones acerca de cómo organizar las tareas, definir las ontologías y las dependencias sociales, aspectos que son modelables dentro de la herramienta.

GAIA:

GAIA es una metodología para el diseño de sistemas basados en agentes cuyo objetivo es obtener un sistema que maximice alguna medida de calidad global (no se llega a detallar cual). GAIA pretende ayudar al analista a ir sistemáticamente desde unos requisitos iniciales a un diseño que, según los autores, esté lo suficientemente detallado como para ser implementado directamente.

En GAIA se entiende que el objetivo del análisis es conseguir comprender el sistema y su estructura sin referenciar ningún aspecto de implementación. Esto se consigue a través de la idea de organización. Una organización en GAIA es una colección de roles, los cuales mantienen ciertas relaciones con otros y toman parte en patrones institucionalizados de interacción con otros roles. Los roles agrupan cuatro aspectos: responsabilidades del agente, los recursos que se le permite utilizar, las tareas asociadas e interacciones.

GAIA propone trabajar inicialmente con un análisis a alto nivel. En este análisis se usan dos modelos, el modelo de roles para identificar los roles clave en el sistema junto con sus propiedades definitorias y el modelo de interacciones que define las interacciones mediante una referencia a un modelo institucionalizado de intercambio de mensajes, como el FIPA-Request. Tras esta etapa, se entraría en lo que GAIA considera diseño a alto nivel. El objetivo de este diseño es generar tres modelos: el modelo de agentes que define los tipos de agente que existen, cuántas instancias de cada tipo y qué papeles juega cada agente, el modelo de servicios que identifica los servicios (funciones del agente) asociados a cada rol, y un Modelo de conocidos, que define los enlaces de comunicaciones que existen entre los agentes.

A partir de aquí se aplicarían técnicas clásicas de diseño orientado a objetos. Sin embargo, esto queda fuera del ámbito de GAIA. Esta metodología sólo buscar especificar cómo una sociedad de agentes colabora para alcanzar los objetivos del sistema, y qué se requiere de cada uno para lograr esto último.

La principal crítica que se puede hacer a GAIA es que se queda a un nivel de abstracción demasiado alto. Según los autores, con ello se consigue desacoplarse de las distintas soluciones de implementación de agentes. Sin embargo, es cuestionable la utilidad de una metodología que genera especificaciones cuya implementación no se llega siquiera a considerar cuando en principio se pretendía llegar a un nivel de detalle fácilmente implementable.

MESSAGE:

MESSAGE Caire y Otros. (2002). es la metodología más reciente de las estudiadas y por tanto trata de integrar resultados de las anteriores. Propone el análisis y diseño del SMA desde cinco puntos de vista para capturar los diferentes aspectos de un SMA: el de Organización, que captura la estructura global del sistema; el de Tareas/Objetivos, que determina qué hace el SMA y sus agentes constituyentes en términos de los objetivos que persiguen y las tareas implicadas en el proceso; el de Agente, que contiene una descripción detallada y extensa de cada agente y rol dentro del SMA; el de Dominio que actúa como repositorio de información (para entidades y relaciones) concernientes al dominio del problema; y el de Interacción, que trata las interacciones a distintos niveles de abstracción.

Estos elementos están presentes en los dos modelos fundamentales que propone MESSAGE: el modelo de análisis y el modelo de diseño. El modelo de análisis se limita a generar modelos a partir de los meta-modelos. El modelo de diseño no llegó a concretarse completamente. Se decidió que el propósito del diseño sería producir entidades computacionales que representen el SMA descrito en el análisis. Por ello, cada artefacto producido en el análisis debería transformarse en una entidad computacional o varias cuyo comportamiento fuera el que se esperaba en el análisis. Esto significa que las entidades del análisis se deberían traducir a subsistemas, interfaces, clases, signaturas de operaciones, algoritmos, objetos, diagramas de objetos y otros.

MESSAGE aporta mejoras en cuanto a conceptos de ingeniería respecto de las alternativas existentes, entre ellas el desarrollo dentro de un paradigma de ingeniería del software (el Proceso Racional Unificado), aportación de métodos para la traducción de entidades de análisis a entidades de diseño y guías para la generación de los modelos.

Sin embargo, los objetivos de MESSAGE no se completaron totalmente. La integración con el Proceso Racional Unificado no fue total, ya que las actividades definidas no se adecuaban a las necesidades reales y no se indicó cómo encajaban dentro de este proceso. Además, faltó trabajo en el estudio de las interdependencias entre los distintos modelos propuestos.

A favor de MESSAGE hay que destacar que ha sido la primera metodología en utilizar una herramienta para soporte del proceso de especificación de SMA de forma visual, como en UML. En cuanto a la implementación, MESSAGE provee guías en cuanto a posibles arquitecturas y componentes a utilizar en esta etapa. Basándose en estas guías y los modelos de análisis y diseño, se realizó manualmente la implementación, lo cual hizo que se detectaran incorrecciones en las definiciones iniciales de los modelos. Esta experiencia es la base de la crítica realizada con anterioridad a ZEUS.

INGENIAS:

El método de desarrollo de SMA propuesto en INGENIAS concibe el SMA como la representación computacional de un conjunto de modelos. Cada uno de estos modelos muestra una visión parcial del SMA: los agentes que lo componen, las interacciones que existen entre ellos, cómo se organizan para proporcionar la funcionalidad del sistema, qué información es relevante en el dominio y cómo es el entorno en el que se ubica el sistema a desarrollar. Para facilitar la representación computacional de los modelos, en este trabajo se asume que la implementación final se realiza parametrizando un armazón software.

Para especificar cómo tienen que ser estos modelos se definen meta-modelos. Un meta-modelo es una representación de los tipos de entidades que pueden existir en un modelo, sus relaciones y restricciones de aplicación. Los meta-modelos que se

describen son una evolución del trabajo realizado en MESSAGE, Caire y Otros. (2002). En MESSAGE se propusieron meta-modelos para representar agentes, organizaciones, el dominio, interacciones, tareas y objetivos. Como se señaló en la sección anterior, el trabajo de MESSAGE es mejorable en cuatro aspectos: la integración de los meta-modelos con las prácticas de ingeniería, un mayor nivel de detalle en los meta-modelos, una mayor cohesión entre los meta-modelos y representación del entorno del sistema.

Por otra parte, INGENIAS, según Gómez (2002) está basada en MESSAGE, Carie y Otros (2002) y emplea meta-modelos y la construcción de modelos usando lenguajes de meta-modelado. Un meta-modelo define primitivas y propiedades sintácticas y semánticas de un modelo y consta de objetos, atributos y relaciones. Durante las fases de análisis y diseño se emplean cinco meta-modelos distintos: (i) meta-modelo de agente, que describe los agentes particulares y sus estados mentales; (ii) meta-modelo de organización, que define cómo se agrupan los agentes, la funcionalidad del sistema y las restricciones existentes sobre el comportamiento de los agentes (iii) meta-modelo de interacción, que detalla cómo se coordinan y comunican los agentes; (iv) meta-modelo del entorno, que define qué existe alrededor del sistema (recursos, aplicaciones); y (v) meta-modelo de tareas y objetivos, que asocia el estado mental del agente con las tareas que ejecuta.

La metodología INGENIAS sigue un proceso iterativo de desarrollo del sistema, según el Rational Unified Process (RUP). De este modo, las fases de análisis y diseño se dividen en tres partes: inicio, elaboración y construcción. En el análisis-inicio, se identifican los casos de uso (por ejemplo, mediante diagramas en UML), se especifica el modelo de organización (identificando los grupos, sus miembros, los flujos de trabajo y los objetivos de la organización) y se modela el entorno (identificando sus aplicaciones y las operaciones a realizar sobre ellas). En el diseño-inicio se genera un prototipo del sistema (por ejemplo, en HTML). Dicho prototipo muestra cómo interactúa el usuario con la aplicación y qué tipo de resultados se obtendrán.

En el análisis-elaboración se completan el resto de modelos. Así, se refinan los casos de uso y se asocian a modelos de interacción y de agente, de modo que se identifican los agentes, sus objetivos, funcionalidades y se especifican sus interacciones mediante diagramas de colaboración. Posteriormente se refinan los modelos de tareas y objetivos (descomponiéndolos en subáreas y subobjetivos y estableciendo relaciones entre sí) y el modelo del entorno (identificando aplicaciones internas). Finalmente se refina el modelo de organización, generando nuevos miembros y descomponiendo los flujos de trabajo.

En el diseño-elaboración se detalla el control de los agentes (asociando estados mentales a la ejecución de acciones). Además, se refina el modelo del entorno respecto a los recursos y la percepción de los agentes así como el modelo de tareas y objetivos, identificando precondiciones y post-condiciones de las tareas. Finalmente, en el análisis-construcción y en el diseño construcción se completa el modelo de organización, estableciendo y refinando las dependencias sociales del sistema respecto a otras organizaciones.

Evaluación de la metodología de desarrollo:

Luego de un estudio de algunas de las metodologías de agente existentes, para este trabajo se eligió la metodología INGENIAS, debido a que tiene un enfoque mas orientado a agente, además posee un proceso de desarrollo robusto, detallado y ha sido probada en desarrollo reales. Así mismo posee una buena y completa técnica de modelado.

A continuación se mostraran las ventajas de la Metodología INGENIAS con respecto a cada una de las otras Metodologías descritas anteriormente de acuerdo a "INGENIAS" (2005):

INGENIAS y MAS-CommonKADS :

MAS-CommonKADS es la metodología más cercana a las líneas principales de INGENIAS. Incorpora la idea de proceso de ingeniería en el sentido de Pressman (1982), y describe con bastante detalle cómo se debe definir el sistema teniendo en cuenta las dependencias entre los modelos. En contra hay que decir que no tiene herramientas de soporte específicas y que presenta problemas a la hora de razonar sobre la especificación de forma automática.

La especificación de SMA que proporciona MAS-CommonKADS detalla la mayoría de aspectos en lenguaje natural. Esta particularidad dificulta el análisis automático de la especificación generada y supone una gran desventaja frente a semiformalismos como UML, soportado por muchas herramientas y con la posibilidad de hacer chequeos para verificar el desarrollo (¿Existen elementos no utilizados? ¿Se ha asociado especificaciones de comportamiento a los casos de uso?). Para lograr lo mismo en MAS-CommonKADS habría que restringir el uso de lenguaje natural o bien incluir formalismos que logren una definición más precisa y menos ambigua del SMA.

En INGENIAS, el problema se salva utilizando meta-modelos como mecanismo de especificación. El desarrollo de meta-modelos está soportado por herramientas que permite el procesamiento automático de los modelos generados. Los meta-modelos en algunos casos profundizan más en el detalle que MAS-CommonKADS. Tal es el caso del meta-modelo de organización, el de tareas y objetivos, o el de agentes.

INGENIAS y BDI:

En INGENIAS, los principios del BDI se hallan presentes de diferentes formas. Los planes, entendidos como secuenciación de tareas, aparecen como flujos de trabajo. Los objetivos han sido introducidos explícitamente, incluyendo su

refinamiento en subobjetivos, el establecimiento de dependencias entre objetivos (inicialmente árboles Y/O) y dependencias con tareas. Su uso se ha extendido a otros modelos como el de organización y el de interacciones, vertebrando la ejecución de tareas e iniciación de interacciones a lo largo de toda la metodología. Por lo tanto, el grado de integración del modelo BDI en la metodología es superior al logrado en la propuesta de Kinny y Georgeff (1997) . Y todo ello dentro de un ciclo de vida de software complejo, como es el Rational Unified Process (RUP). Además, se ha logrado la independencia total de implementación dotando a la metodología de un procedimiento para parametrizar con los modelos generados armazones software escritos en cualquier lenguaje.

INGENIAS y MASE:

En INGENIAS, se persigue un proceso de desarrollo similar al seguido mediante otras herramientas, como Rational Rose, TogetherJ o Paradigm+. En estas herramientas, el usuario tiene libertad para elaborar diagramas incompletos o generar sus propias vistas del sistema, tomando elementos de diferentes modelos e incorporándolos a un nuevo diagrama. También se intenta separar de UML en el sentido de no repetir las mismas soluciones. La forma en que se tratan las interacciones en INGENIAS es un buen ejemplo de ello. Las interacciones buscan generalizar diferentes alternativas existentes, como los MSC, diagramas de secuencia o colaboración y diagramas de protocolo Agente Basado en Lenguaje Unificado de Modelado (AUML). Aparte de la generalización, las interacciones se integran completamente dentro de otros modelos como un elemento más. De hecho, la iniciación de interacciones se representa como el producto de ejecutar una tarea.

Otro aspecto tomado en cuenta en INGENIAS es el proceso de generación de los modelos, esto es, qué actividades están involucradas en su producción. El resultado es un conjunto estructurado de actividades detalladas que guiarán a futuros usuarios de la metodología en su utilización. Las actividades se enmarcan en diferentes flujos de

trabajo para cada modelo, con lo que se facilita una futura integración de la metodología en entornos automatizados de gestión de proyectos software como el Entorno de Creación de Servicios de Telefónica I+D.

INGENIAS y ZEUS:

La postura de INGENIAS es que las herramientas de soporte no tienen que condicionar la metodología y que de hecho han de ser independientes. La independencia se consigue usando meta-modelos como elemento de construcción. Ello facilita el portar la metodología a diferentes herramientas, ya que cualquier herramienta que soporte meta-modelado podría servir como herramienta soporte de desarrollo.

Otra ventaja de los meta-modelos es que facilitan la evolución de la metodología. Tanto ZEUS como MaSE tendrían que cambiar en gran medida sus herramientas asociadas, para, por ejemplo, incluir el meta-modelo de organización de este trabajo. Sin embargo, el paso inverso, incluir elementos de MaSE o ZEUS en meta-modelos, no supone un gran esfuerzo.

INGENIAS y GAIA:

Como solución genérica, en INGENIAS, se proporciona un procedimiento con el que se facilita el salto a la implementación del sistema. La implementación se plantea como la parametrización de un armazón software utilizando los modelos que componen la especificación.

Otro aspecto discutible es el uso combinado de fórmulas lógicas con fichas de documentación clásicas en ingeniería del software. En GAIA parece asumirse que poniendo fórmulas lógicas se consigue una mejor comprensión del problema y una mayor exactitud. El problema de estas fórmulas, aparte de su comprensión, es su

definición. ¿De qué sirve establecer las precondiciones de una tarea con un conjunto de predicados cuya semántica y existencia no está definida en ningún sitio?

Para salvar este problema, aparte de dejar libertad al que quiera utilizar fórmulas lógicas, se ofrecen representaciones donde los términos que componen la representación son los propios elementos del meta-modelo. Por ejemplo, se puede hablar de las cualidades de un agente con el que se quiere interactuar utilizando un modelo de agente que represente los roles que debe tener, los objetivos asociados y estado mental en el que se supone que debe estar el agente colaborador.

Como en MaSE, además, se comete el error de obviar las distintas dependencias entre los modelos propuestos, lo cual es fundamental a la hora de proponer un proceso que dé cómo salida la especificación del sistema.

En INGENIAS las dependencias entre los distintos meta-modelos se revisan independientemente. La mayoría se refiere a que al introducir ciertas entidades, hay que definir aspectos adicionales en otros modelos. Por ejemplo, al crear un objetivo, siempre hay que asociar una tarea o tareas que permiten alcanzarlo. Si el objetivo se identifica dentro de un modelo de agente, entonces, se necesita que en un modelo de objetivos y tareas se indique qué tarea o tareas deben ejecutarse

En INGENIAS el meta-modelo de organización se ve respecto del SMA como el equivalente a la arquitectura del sistema de un sistema convencional. Sirve para definir a alto nivel cómo se organizan los elementos del sistema para hacer posible los objetivos comunes a los agentes que participan en la organización.

INGENIAS y MESSAGE:

En INGENIAS, se plantea la evolución a lo largo del ciclo de vida del software de los modelos generados. El paso de una etapa a otra está marcado por el nivel de

detalle alcanzado en cada modelo. Así, las interacciones inicialmente pueden detallarse con diagramas de colaboración para luego concretarse en el diseño con otros tipos de diagramas que alcancen más detalle en aspectos como la motivación de la interacción o actos del habla empleados durante el proceso. El paso a implementación, como se ha comentado antes, se ha generalizado en forma de proceso de parametrización de armazones software. Esta forma de implementación es una evolución del trabajo de MESSAGE, donde se proponían arquitecturas y componentes adecuados para esta tarea.

Los meta-modelos en general se han modificado para integrar resultados de investigación tales como planificación de tareas, el modelo BDI, la estructuración de elementos de la comunidad o el uso de tareas. De forma similar a MAS-CommonKADS se ha estudiado el dominio de aplicación de cada meta-modelo para que se puedan aplicar los resultados correspondientes.

También se ha incluido un nuevo meta-modelo, el de entorno. MESSAGE no tenía en cuenta lo que rodeaba la aplicación, por lo cual la inclusión de elementos como servicios del sistema, recursos o aplicaciones que no fueran agentes, eran difíciles de tratar. En INGENIAS, el meta-modelo de entorno permite incluir este tipo de elementos de forma coherente. De hecho, la percepción de los agentes se expresa en función de estos elementos. Así, se puede representar que un agente de interfaz se conecte a una aplicación existente.

INGENIAS Development Kit:

INGENIAS como evolución de las ideas de MESSAGE. INGENIAS profundiza en los elementos mostrados en el método de especificación, en el proceso de desarrollo, además de incorporar nuevas herramientas de soporte y ejemplos de desarrollo. INGENIAS, como MESSAGE, define un conjunto de meta-modelos (una descripción de alto nivel de qué elementos tiene un modelo) con los que hay que

describir el sistema. Los meta-modelos indican qué hace falta para describir: agentes aislados, organizaciones de agentes, el entorno, interacciones entre agentes o roles, tareas y objetivos. Estos meta-modelos se construyen mediante un lenguaje de meta-modelado, el GOPRR (Graph, Object, Property, Relationship, and Role). En la construcción de estos meta-modelos se integran resultados de investigación en forma de entidades y relaciones entre entidades. La instanciación de estos meta-modelos produce diagramas, los modelos, similares a los que se usa en UML, con la diferencia de que estos diagramas se han creado exclusivamente para definir el sistema multi-agente.

El proceso de instanciación de los meta-modelos no es trivial. Existen muchas entidades y relaciones a identificar, además de dependencias entre distintos modelos. Por ello, INGENIAS define un conjunto de actividades cuya ejecución termina en un conjunto de modelos. Estas actividades a su vez se organizan siguiendo un paradigma de ingeniería del software, el Proceso Unificado. La ejecución de actividades para producir modelos se basa en la herramienta INGENIAS IDE, una herramienta para modelado visual.

INGENIAS Development Kit (IDK), es un editor de modelos, donde se pueden diagramar los modelos que conforman la metodología INGENIAS, realiza la integración de modelos y la integración con agentes en desarrollo.

El editor presenta una nomenclatura para representar una serie de relaciones que se realizan en cada uno de los diagramas o modelos. Así como también existe una notación para realizar el diseño de dichos diagramas, por tal razón es importante conocerlas.

El nombre asociado a las relaciones obedece a unas reglas sencillas mostradas. Se trata de hacer que el nombre de la relación sea precedido por un conjunto de letras que denote su procedencia, como el flujo de trabajo (WF), meta-modelo de agente (A), interacción (I), unidad de interacción (UI), modelos de tareas y objetivos (GT),

relaciones sociales (AGO), organización (O) o el entorno (E). En la figura 3, se muestra la nomenclatura antes descrita

Figura 3

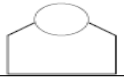
Nomenclatura

IdentificadorRelacion::= RelacionFlujoTrabajo | RelacionAgente | RelacionInteraccion | RelacionUnidadInteraccion | RelacionMetaTarea| RelacionSocial | RelacionOrganizacion | RelacionEntorno
RelacionFlujoTrabajo::= **WF** Identificador
RelacionAgente ::= **A** Identificador
RelacionInteraccion::= **I** Identificador
RelacionUnidadInteraccion ::= **UI** Identificador
RelacionMetaTarea::= **GT** Relacion
RelacionSocial ::= **AGO** Relacion
RelacionOrganizacion ::= **O** Relacion
RelacionEntorno ::= **E** Relacion

Los diagramas, que se presentan a lo largo de este trabajo utilizan una notación. Para poder comprender los diagramas debe tomar en cuenta la notación de cada componente del editor. Las figuras 4,5,6 y 7 muestran la notación de los componentes para diagramar en IDK

Figura 4

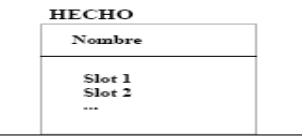
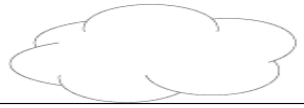
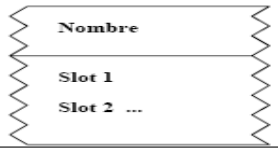
Notación de Componentes para Diagramar en IDK

	**Objetivo**. Se etiqueta con el nombre del objetivo
	**Rol**. Se etiqueta con el nombre del rol.
	**Procesador de estado mental**. Se etiqueta con el nombre del procesador.
	**Gestor de estado mental**. Se etiqueta con el nombre del gestor.
	**Agente**. Se etiqueta con el nombre del agente.

Fuente: herramienta de ayuda del IDK, traduccion Bieliukas (2010)

Figura 5

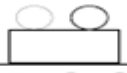
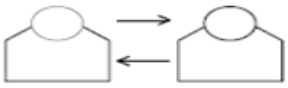
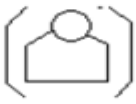
Notación de Componentes para Diagramar en IDK (Continuación)

	**Hecho.** Se etiqueta con el nombre del hecho y los nombres de los slots identificados.
	**Creencia.** Se etiqueta con el nombre de la evidencia e información acerca de qué es lo que se está aceptando como cierto.
	**Evento.** Se etiqueta con el nombre del evento y los nombres de los slots identificados.

Fuente: Obtenida de la herramienta de ayuda del IDK, traduccion Bieliukas (2010)

Figura 6

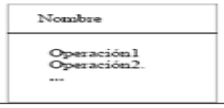
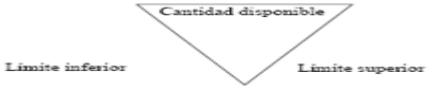
Notación de Componentes para Diagramar en IDK (Continuación)

	**Grupo.** Se etiqueta con el nombre del grupo.
	**Organización**. Se etiqueta con el nombre de la organización.
	**Flujo de trabajo.** Se etiqueta con el nombre del flujo.
	Interacción. Se etiqueta con el nombre de la interacción y su naturaleza, como coordinación, planificación o negociación.
	**Consulta de entidades autónomas**. Se etiqueta con nombres concretos de agentes existentes o expresiones que denotan agentes existentes.

Fuente: Obtenidad de la herramienta de ayuda del IDK (2010)

Figura 7

Notación de Componentes para Diagramar en IDK (Continuación)

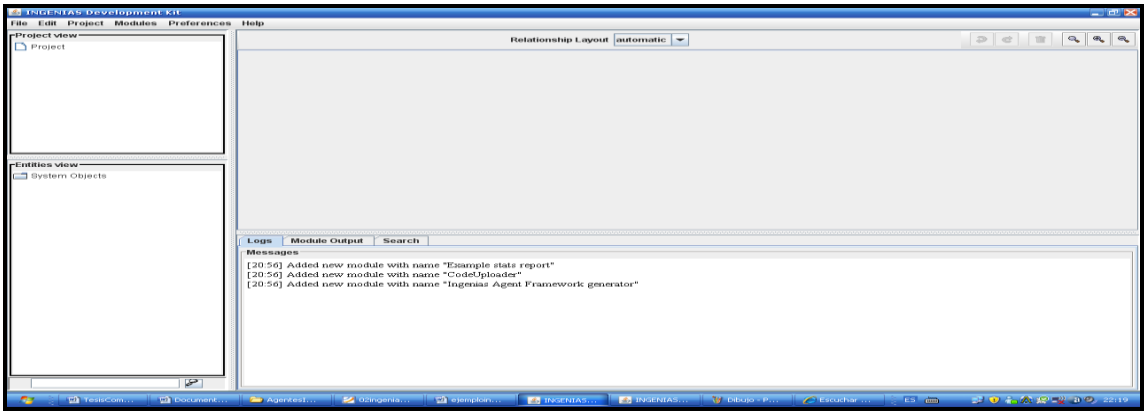
	**Unidad de interacción.** Se etiqueta con el nombre de la unidad y el acto del habla al que hace referencia, como *request*, *inform*, o *not-understood*.
	**Tarea.** Se etiqueta con el nombre de la tarea.
	**Aplicación.** Se etiqueta con el nombre de la aplicación y las operaciones soportadas.
	**Recurso.** Se etiqueta con el nombre del recurso, la cantidad disponible del mismo, el límite inferior y superior admisibles. Por debajo o encima de estos límites, el recurso se deshabilita.

Fuente: Obtenida de la herramienta de ayuda del IDK, traduccion Bieliukas (2010)

Al trabajar con el editor IDK aparecerá una pantalla como se muestra en la siguiente figura, la cual es el ambiente de trabajo para desarrollar el diseño de los diagramas.

Figura 8

Pantalla Principal del editor IDK

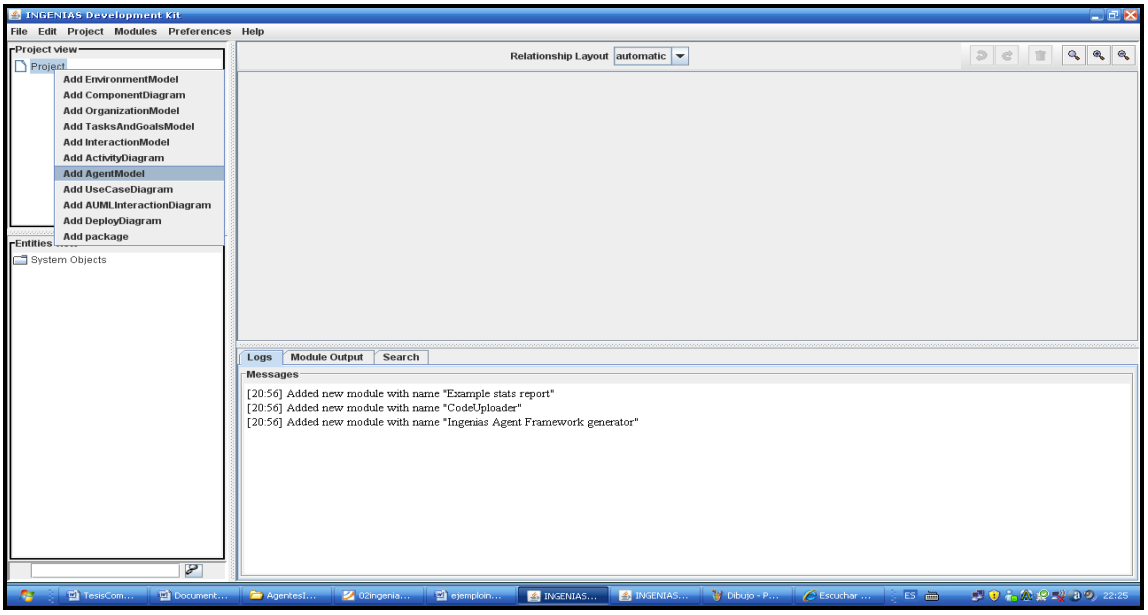


Fuente: Bieliukas (2010)

Ahora bien, se procede ha describir como funciona el editor, utilizando como ejemplo el diseño de uno de los diagramas. Luego de que se inicia el editor y este muestra la pantalla principal, debe posicionarse en el recuadro superior izquierdo y colocar el ratón en donde dice Project, con el botón derecho del ratón se selecciona el diagrama que se quiere diseñar. Así como lo muestra la figura 9, donde se muestra la selección de uno de los diagramas que se van a diseñar.

Figura 9

Paso para seleccionar el diagrama que se va a diseñar, por ejemplo Modelo de Agente

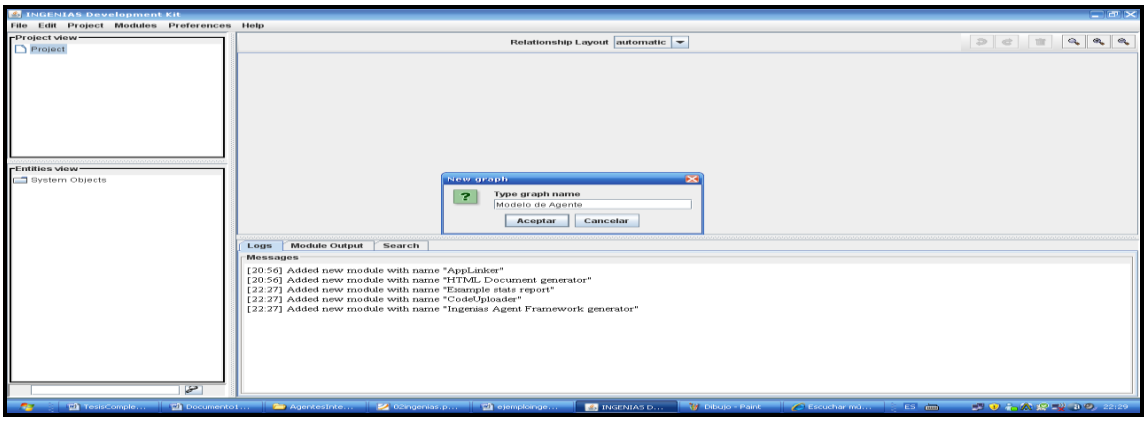


Fuente: Bieliukas (2010)

A continuación se muestra una ventana donde se debe colocar el nombre del diagrama que se va a diseñar y luego se selecciona el botón aceptar, tal como lo muestra la siguiente figura en la cual se refleja el recuadro donde se debe colocar el nombre del diagrama que se va a diseñar.

Figura 10

Colocarle el nombre al diagrama

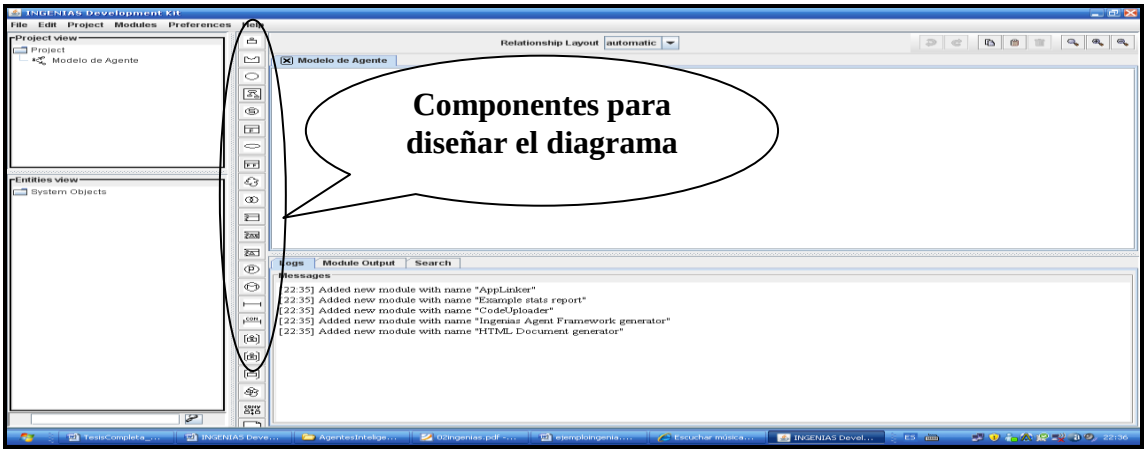


Fuente: Bieliukas (2010)

Posteriormente se mostraran todos los componentes necesarios para el desarrollo del diseño del diagrama, es importante señalar que los componentes varían dependiendo del diagrama que se desea desarrollar. Lo anteriormente descrito se muestra en la figura 11.

Figura 11

Pantalla que muestra los componentes para diseñar el diagrama

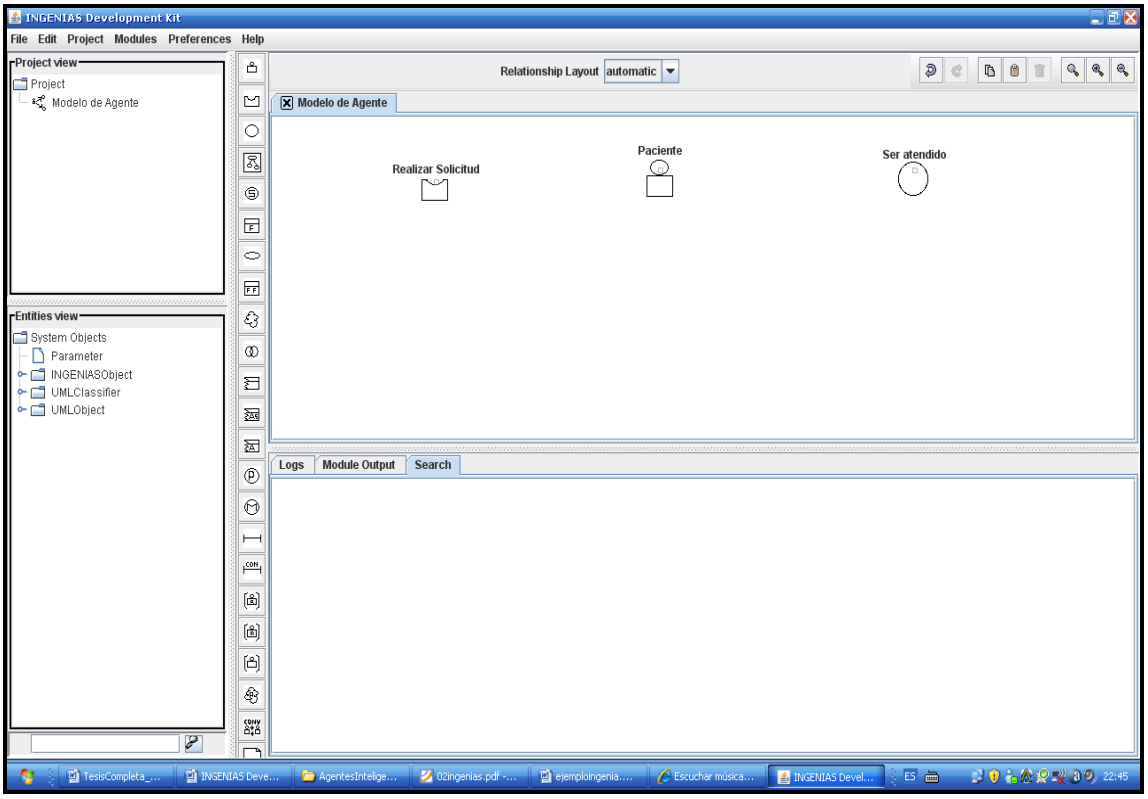


Fuente: Bieliukas (2010)

Ahora se procede a diseñar el diagrama, cada componente se coloca seleccionando con el ratón el componente que se necesita, este se ubica en la pantalla de diseño, en este caso se van a escoger un agente, un objetivo y dos roles. Observar la figura 12 que muestra el proceso descrito.

Figura 12

Selección de Componentes para diseñar el diagrama

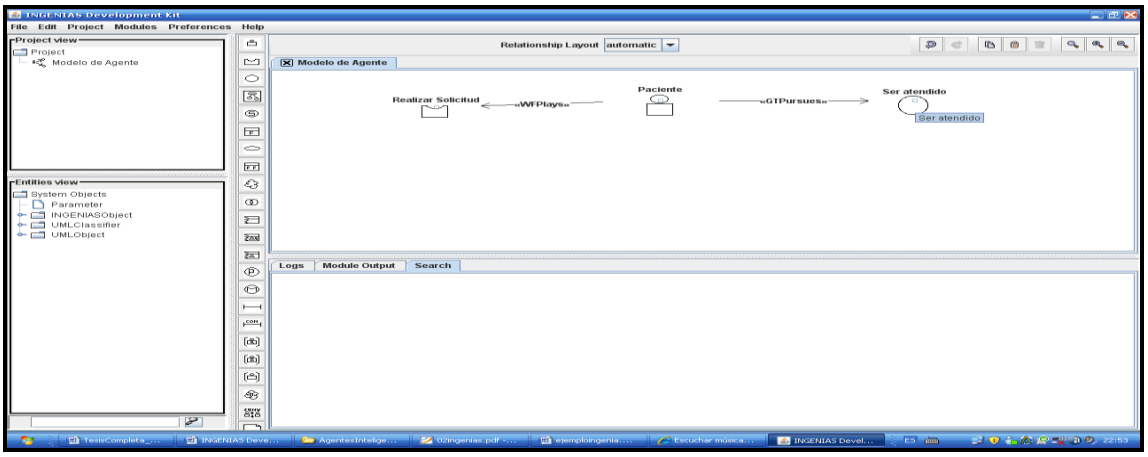


Fuente: Bieliukas (2010)

El próximo paso es realizar las relaciones entre cada componente, para que esto ocurra se debe seleccionar con el ratón un cuadro que tiene en el centro cada componente, se selecciona uno y luego se arrastra hasta el cuadro del centro del otro componente que se quiere relacionar. Ver figura 13

Figura 13

Relaciones entre los componentes

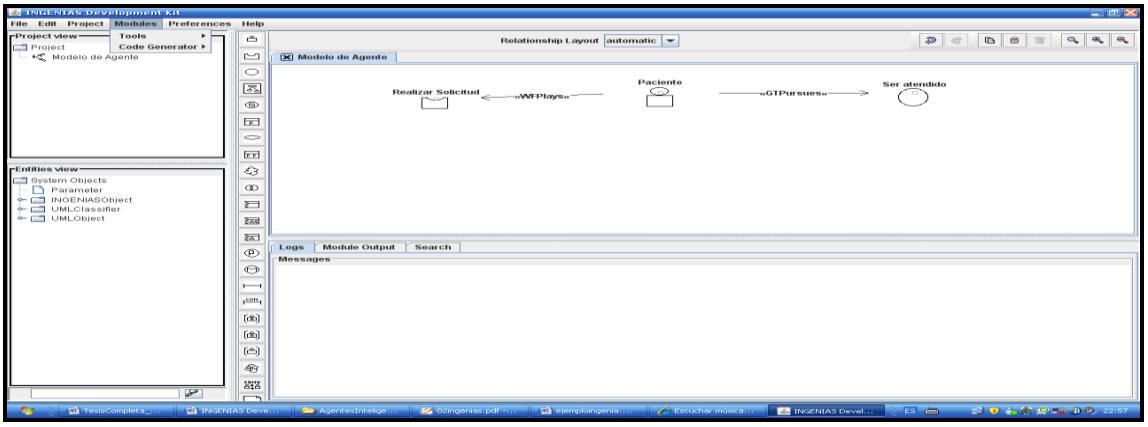


Fuente: Bieliukas (2010)

A continuación las siguientes imágenes, presentan los pasos para generar el código fuente que genera el editor. En las figuras 14,15,16,17 y 18 se reflejan los pasos que se deben realizar para que se genere el código fuente de los diagramas diseñados.

Figura 14

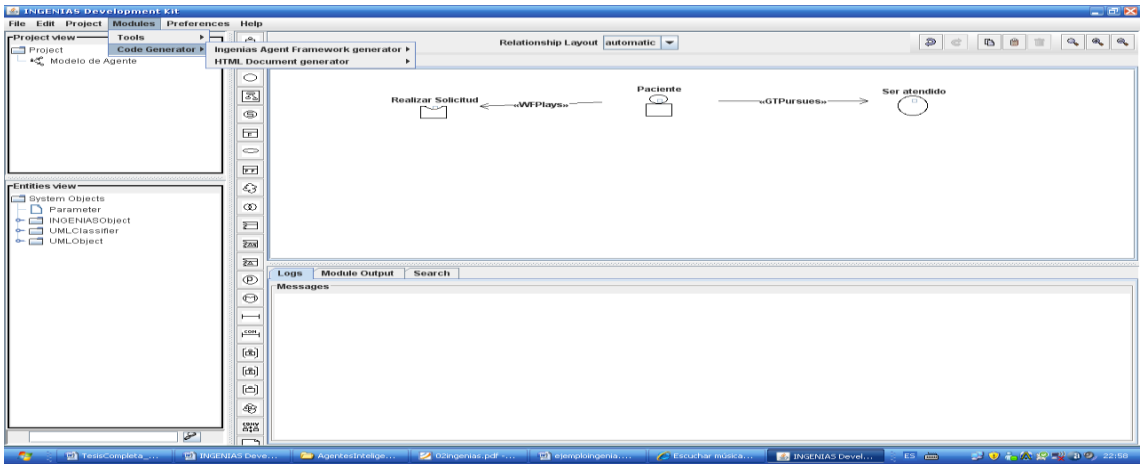
Paso 1 seleccionar en el menú de la parte superior la opción de Modules



Fuente: Bieliukas (2010)

Figura 15

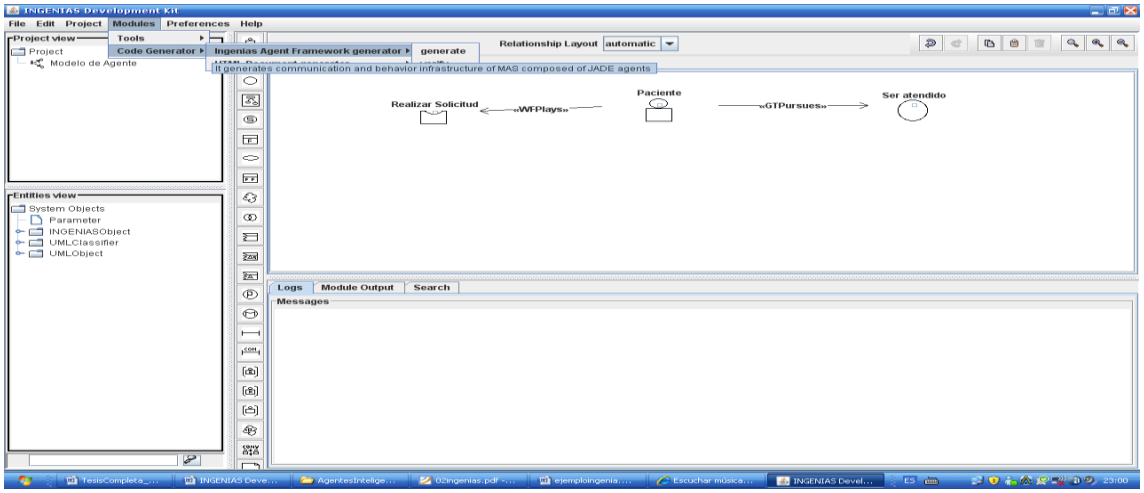
Paso 2 Seleccionar la opción de Code Generator



Fuente: Bieliukas (2010)

Figura 16

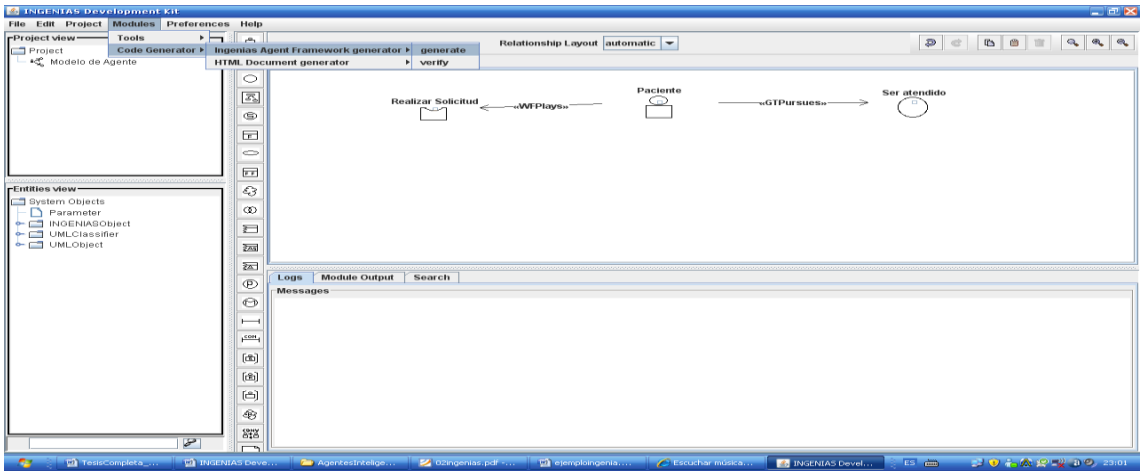
Paso 3 se selecciona la opción la opción INGENIAS Agent Framewor Generator



Fuente: Bieliukas (2010)

Figura 17

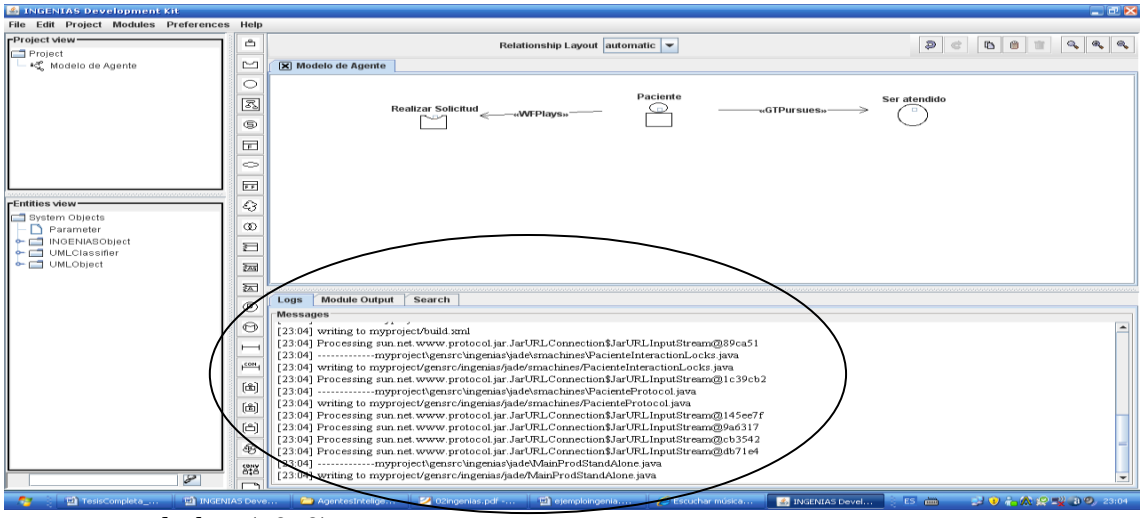
Paso 4 Seleccionar la Opción Generate



Fuente: Bieliukas (2010)

Figura 18

Paso 5 se muestra en la pantalla de log que el código se generó correctamente



Fuente: Bieliukas (2010)

JADE (Java Agent DEvelopment Framework):

Jade es una librería que se instala sobre máquinas virtuales Java y constituye una plataforma distribuida con todos los elementos necesarios para que máquinas virtuales en diferentes computadoras físicas interconectadas soporten la ejecución de los agentes del sistema. Jade tiene una interfaz gráfica que permite monitorear el sistema, incorpora varios mecanismos de comunicación y modelado de información y por supuesto la clase agente con otras clases para modelar comportamientos diferentes. Jade es una plataforma con mucha acogida entre sectores industriales y académicos.

Jade es básicamente dos cosas:

- Una plataforma: que permite "VIVIR" y "CONVIVIR" a los agentes dentro de ella. Es decir, proporciona el entorno necesario para la ejecución de estos agentes simultáneamente y los canales para su comunicación y cooperación.
- Un conjunto de herramientas con el que realizar el desarrollo de agentes y sistemas multiagente de forma más sencilla, rápida y efectiva. Labor tediosa.

Totalmente realizado en Java. (Portabilidad y Movilidad)

Un Agente en JADE:

Un agente en JADE es una instancia de una clase definida por el usuario la cual extiende de la clase Agent (jade.core.Agent). Importante recalcar que cada agente JADE se implementa como un único hilo de ejecución.

Como Trabajar con Jade:

Antes de empezar a trabajar con JADE,se debe tener instalada la maquina virtual de java (JVM). Si no se tiene, debe instalar para luego obtener lo

necesario para usar el framework. Después de instalar la JVM o si se encuentra procedemos a lo siguiente:

Descargue de la pagina http://jade.tilab.com/download.php el archivo "JADE-all-3[1].6.zip" haciendo clic en el link jadeAll.zip que contiene todo lo referente a los agentes JADE, guarde en el disco duro(C:\) y descomprímalo, al descomprimirse aparecerán los archivos siguientes, JADE-bin-3.6.zip, JADE-doc-3.6.zip, JADE-examples-3.6.zip, JADE-src-3.6.zip, el primero se debe descomprimir porque es necesario ya que contiene todos los archivos referentes para poder trabajar con JADE, es decir, contiene los códigos compilados y listos para ser interpretados, los otros son el API, ejemplos y el fuente sin compilar de las clases de JADE.

Cuando se descomprime el archivo "JADE-bin-3.6.zip", crea una carpeta llamada jade, dentro de esa carpeta se crea otra llamada "lib" donde están los archivos .jar que se deben agregar al CLASSPATH, usando el siguiente comando en Windows. Así tenemos la posibilidad de trabajar con JADE, desde la consola o desde cualquier herramienta de trabajo para JAVA

CAPITULO III

MARCO METODOLÓGICO

Naturaleza de la Investigación

Esta investigación forma parte de un trabajo de grado que se define en el Manual para presentación del trabajo conducente al grado académico de Especialización, Maestría y Doctorado de la UCLA (2002) como el resultado de la actividad de investigación del estudiante, quien profundiza en el conocimiento de algún tema especifico o área de estudio, demostrando dominio de la metodología científica acorde con la naturaleza del problema objeto de investigación.

Para el caso de la investigación se presenta a profundidad la problemática que se presenta en el servicio de Emergencia del Hospital Central Universitario Antonio María Pineda, en cuanto al congestionamiento de la misma y la metodología científica planteada es el diseño de un prototipo de un sistema multiagente para apoyar la gestión del servicio de emergencia.

Así mismo la investigación representa un proyecto factible debido a que se pretende resolver un problema real, en el cual se estudia y analiza la situación actual del problema. Presentar una solución a través de un diseño de una propuesta , esto es señalado en el Manual para presentación del trabajo conducente al grado académico de: Especialización, Maestría y Doctorado de la UCLA (2002) donde define un proyecto factible como “Una propuesta sustentada en un modelo viable para resolver un problema práctico planteado, tendente a satisfacer necesidades institucionales o

sociales, que pueden referirse a la formulación políticas, programas, tecnologías, métodos y procesos". Se apoya en la investigación documental y de campo.

De acuerdo con Cázares y Otros (2000), La investigación documental depende fundamentalmente de la información que se recoge o consulta en documentos, entendiéndose este término, en sentido amplio, como todo material de índole permanente, es decir, al que se puede acudir como fuente o referencia en cualquier momento o lugar, sin que se altere su naturaleza o sentido, para que aporte información o rinda cuentas de una realidad o acontecimiento.

Las fuentes documentales pueden ser, entre otras: documento escritos, como libros, periódicos, revistas, actas notariales, tratados, encuestas y conferencias escritas; documentos fílmicos, como películas, diapositivas, fílmicas; documentos grabado, como discos, cintas y casetes, incluso documentos electrónicos como páginas web.

Según Cázares y otros (2000), la investigación de campo es aquella en que el mismo objeto de estudio sirve como fuente de información para el investigador. Consiste en la observación, directa y en vivo, de cosas, comportamiento de personas, circunstancia en que ocurren ciertos hechos; por ese motivo la naturaleza de las fuentes determina la manera de obtener los datos.

Las técnicas usualmente utilizadas en el trabajo de campo para el acopio de material son: la encuesta, la entrevista, la grabación, la filmación, la fotografía, entre otras, de acuerdo con el tipo de trabajo que se está realizado, puede emplearse una de estas técnicas o varias al mismo tiempo.

Entre las mejoras que se plantea es aportar una solución a la situación planteada por medio del diseño de un prototipo, el cual podrá dar un recepción más eficiente, indicándole al paciente a donde debe dirigirse de acuerdo a los síntomas que presenta, por otra parte le brinda a los profesionales de la medicina información previa sobre el paciente de esta forma ya podrá conocer los síntomas que presenta el paciente a la

hora de ser atendido, a su vez se podrá llevar el control de los pacientes en cada lugar de referencia.

Fases del Estudio

Fase 1: Diagnóstica

Procedimiento

En lo esencial la propuesta planteada, está representada por el diseño de un sistema multiagente usando metodología INGENIAS para la gestión de servicio de emergencia médica hospitalaria específicamente para el Hospital Central Antonio María Pineda.

En este sentido, es importante comenzar hablando sobre los objetivos específicos planteados en el capítulo I en el mismo orden. Comenzando por el que plantea realizar un diagnóstico de la situación actual del servicio de emergencia del Hospital Central Antonio María Pineda, para lograr este objetivo se realizaran dos tipos de investigaciones:

Investigación de Campo en la cual se aplicaron técnicas de recolección de datos, como visitas, llamadas y entrevistas a los siguientes profesionales: Coordinador del Postgrado de Emergencia, Secretaria del jefe del servicio de emergencia, coordinadora de estadística del hospital, coordinadora de enfermería del servicio de emergencia, médico cirujano egresado de la UCLA y estudiante de medicina.

Investigación Documental representada por un cuaderno donde se llevan los datos de los pacientes que acuden al servicio de emergencia, estadística del número de pacientes atendidos en un año, de la cual se calculó el porcentaje de los pacientes diario.

Así mismo, después de la aplicación de las investigaciones se realizó un análisis de los datos obtenidos y se efectuará un diagnóstico de la problemática que se

presenta en dicha sala de emergencia. Proponiendo para la solución el diseño de un sistema multiagente, donde las personas que fueron entrevistadas estuvieron de acuerdo con la propuesta y les pareció una buena opción para ayudarlos a mejorar la gestión de servicio.

Sobre la base de las consideraciones anteriores, se procederá a trabajar con el objetivo específico, que se trata de modelar el prototipo del sistema multiagente que va a ayudar a disminuir la afluencia de los pacientes en el servicio de emergencia del hospital, basándose en la metodología de agentes ingenia. En efecto luego de modelar el sistema se procede a diseñarlo.

Fase 2: Estudio de Factibilidad

Factibilidad Operacional

El diseño de un sistema multiagente usando metodología INGENIAS para la gestión de servicio de emergencia médica hospitalaria, busca ser un apoyo que pueda servir a mejorar la recepción de los pacientes en dicho servicio, ayudando a evitar el congestionamiento.

Operativamente esta investigación se considera factible debido a que:

- Satisface las necesidades relacionadas con la gestión de servicio en la recepción del servicio de emergencia del Hospital Central Universitario Antonio María Pineda.

- El Hospital necesita controlar y administrar el acceso de los pacientes que acuden al servicio de emergencia, así como también tener información de los mismos. Presentar un diseño del sistema multiagente significa una mejora al servicio de recepción de los pacientes, además los profesionales de la medicina podrán tener una mayor información de los pacientes y podrán realizar mejor sus estudios de los mismos. Se puede decir surgirán muchos beneficios.

- Se le proveerá capacitación del manejo del sistema mediante un curso de capacitación sobre los conceptos, ventajas y funcionamiento del Sistema.

- Se contara con manuales del usuario, el cual explicara el funcionamiento del mismo.

- Se le informara a la sociedad sobre el uso del sistema y los beneficios que este provee, así como también la forma de darle uso antes de que este sea implementado.

Factibilidad Técnica

Se considera técnicamente factible debido a que cuenta con la tecnología, información y equipo necesario para la elaboración del proyecto**.**

Para el diseño del sistema se cuenta con un equipo el cual posee las siguientes características: una computadora portátil, Intel Celeron M procesador 15000 MHz 1.5 GHz, 480 MB RAM disco duro de 40 GB. Para la parte del Software se utilizo como herramienta de diseño IDK, Java usando las librerías de Jade para la simulación de los agentes.

Cabe destacar, que desarrollar en software libre aplica a los principios de libre acceso a las fuentes de conocimiento que propugna la ciencia al ámbito del software. Además, se puede compartir el software creado bajo sus directrices pero si no tiene a nadie con quien compartirlo no deja de ser software libre.

El Software Libre, como todo trabajo colectivo, requiere de cierta masa crítica para alcanzar un grado de madurez deseable. El software es un componente vital en las nuevas tecnologías y su fiabilidad es requisito indispensable para un uso en cualquier escala, es por ello que se selecciona desarrollar este sistema bajo esta plataforma.

Se cuenta con manuales técnicos, Internet, a su vez se tiene un recurso humano con conocimientos suficiente sobre el funcionamiento del diseño del sistema. Brindando así la ejecución de requerimientos exigidos por el usuario.

Factibilidad Económica

De acuerdo a los requerimientos especificados en la factibilidad técnica, en relación al software que se utilizará y al equipo, tanto técnico como humano, necesario para la elaboración del sistema, se establece la siguiente valoración de

En la actualidad según el Colegio de Ingenieros de Venezuela, un programador devenga (aproximadamente) un sueldo de 1.300,00 bolívares y el total de líneas de código estimadas son de 3020, así mismo tomando un estándar de productividad media para el desarrollo de un sistema es de 750 LDC/ por mes; podría estimarse el costo aproximado por línea de código en:

CostoEstimado * LDC = 500,00 / 750 =0, 666 Bs / LDC

Si el software está desarrollado en 3020 LDC; entonces el costo total del sistema será:

CostoTotalCodigo = (CostoEstimado / LDC)* KLDC
CostoTotalCodigo = (0,666 Bs/LCD) * 3020LDC
CostoTotalCodigo = 2011,32

Por tanto, el costo arrojado por el trabajo del programador es: 2011,32 Bs.

Seguidamente se presenta una aproximación del costo actual de un computador en el mercado nacional.

Cuadro 6:

Costo aproximado de un PC.

Cantidad	Descripción	Costo (Bsf)	Total (Bsf)
1	Pentium Dual	1.900,00	1900,00

Fuente: Bieliukas (2009)

Fase 3: Diseño de la Propuesta

Esta fase contiene los distintos pasos a realizar, que permiten ir definiendo y reafirmando, las especificaciones del sistema multiagente. En lo que concierne, para el desarrollo del diseño detallado del sistema, se realizara la implementación de la metodología de agentes INGENIAS donde se crearan los modelos, las herramientas y el ciclo de vida del sistema.

En lo que respecta al desarrollo del prototipo del sistema estará conformado por distintos módulos los cuales son: Registro de los usuarios (médicos y pacientes), Emergencia (este se encargara de registrar las enfermedades que se consideran emergencias para ser atendidas en el servicio), Referir (es el que interpretara la enfermedad del paciente de acuerdo a los síntomas que este indique y a su vez lo refiere al sitio que le corresponde, si el sitio que le corresponde es el servicio de emergencia este le informará al paciente la prioridad en la cual será atendido de acuerdo a la emergencia que presente), Consulta (es el que mostrará la información que el usuario requiera).

CAPITULO IV

Análisis de los Resultados

Estructura del Prototipo del Sistema Multiagente

En el presente capitulo se describe detalladamente el desarrollo del prototipo del Sistema multiagente, para la gestión de servicios de emergencia medica hospitalaria usando metodología INGENIAS, el cual tiene como objetivo principal facilitar a los pacientes y al equipo de profesionales que laboran en el servicio de emergencia del Hospital Central Universitario "Antonio María Pineda", la atención del mismo. Es importante señalar que actualmente el servicio no cuenta con recursos que sirvan como herramienta automatizada a la hora de atender a los pacientes.

Para la realización del análisis y diseño del prototipo del sistema multiagente para la gestión de servicios de emergencia medica hospitalaria, se seleccionó la metodología INGENIAS. INGENIAS modela el sistema multiagente como, la representación computacional de un conjunto de modelos. Cada uno de estos modelos muestra una visión parcial del sistema: los agentes que lo componen, las interacciones que existen entre ellos, como se organizan para proporcionar la funcionabilidad del sistema, que información es relevante y como es el entorno en que se ubica el sistema a desarrollar.

A continuación, se mostraran los modelos de cada una de las tres fases (Inicio, Elaboración y Construcción), estos fueron realizados para obtener una visión detallada del diseño del prototipo del sistema multiagente para la gestión de servicios de emergencia medica hospitalaria. Cabe destacar que estos modelos fueron diseñados bajo una herramienta de modelado denominada "INGENIAS Development

Kit IDK", la misma se encarga de crear y modificar modelos de sistemas multiagentes.

Descripción de la aplicación de las Fases de la Metodología INGENIAS

De acuerdo con "INGENIAS" (2005), la metodóloga INGENIAS se encuentra conformada por tres fases las cuales son (Inicio, Elaboración y Construcción), por tal razón a continuación se describen cada una de esas fases con los diagramas que les corresponden:

Fase 1. Inicio:

Esta fase consiste en analizar, la arquitectura del sistema mediante un modelo de organización y el modelo de interacción**.** Así mismo para la parte del diseño se requiere modelar un prototipo con herramientas de prototipado , en este caso como se menciono anteriormente se utilizó la herramienta "INGENIAS Development Kit IDK". Así mismo a continuación se muestra la descripción y el modelado del modelo de organización para el servicio de emergencia del HCUAMP:

Modelo de Organización:

Este modelo es comparable con la estructura del sistema multiagente. En este modelo se describen los departamentos y roles que se encuentran presentes en el sistema. Aplicando esta definición, se describe el prototipo del sistema multiagente para la gestión de servicios de emergencia médica hospitalaria, desde la perspectiva organizacional, dividiéndolo en varios departamentos los cuales se mencionan a continuación:

- **Recepción:** En este departamento se realiza el ingreso del paciente para su atención, por lo que el paciente ingresa al servicio realizando una solicitud para ser atendido. A continuación se describen los roles que conforman el departamento:

1. **Realizar una Solicitud:** consiste en que los pacientes solicitan ser atendido en el servicio.

2. **Asignar Turno:** Este se encarga de analizar las causas por las que el paciente realiza la solicitud, asignándole un turno dependiendo la prioridad en la que se encuentra.

- **Sala de Curas:** Este departamento consiste en recibir al paciente cuando le corresponde el turno, en el mismo se encuentran los bachilleres que estudian medicina, médicos residentes y enfermeras, allí se atiende al paciente, examinándolo, medicándolo si es necesario y diagnosticándolo para referirlo, es decir, a sala de observaciones, sala de trauma shock, ambulatorio o a su casa dependiendo de dicho diagnostico. Así mismo se realiza el registro de los pacientes atendidos. A continuación se presentan los roles del departamento:

    1. **Examinar:** Consiste en que los bachilleres y los residentes se encargan de examinar al paciente, para realizar el diagnostico del mismo.

    2. **Registrador:** Se encarga de registrar los datos del paciente y agregar el diagnostico del paciente, los medicamentos que le administraron y a donde fue referido.

    3. **Medicar:** Se encarga de que las enfermeras le administren al paciente los medicamentos indicados por los residentes.

    4. **Referir:** Este rol consiste en que los residentes se encargar de referir al paciente al lugar que le corresponda dependiendo al diagnostico que presenta (Sala de Observaciones, Sala de Trauma Shok, Ambulatorio, Casa o a la Morgue).

- **Sala de Trauma y Shok:** En este departamento se encuentran aquellos pacientes que ingresan al servicio por accidentes cerebro vasculares, es un departamento de mucho cuidado tienen a enfermeras constantemente

pendiente de esos pacientes, y el acceso al mismo es restringido. Posee los siguientes roles:

1. **Recibir al Paciente:** Este se encarga de realizar el ingreso al paciente en la sala, de prepararlo y ubicarlo en la cama que corresponde con sus respectivos equipos médicos.

2. **Atender al Paciente:** El mismo consiste en revisar cual es el diagnóstico que presenta el paciente, volver a examinarlo y darle los cuidados que sean necesarios.

3. **Suministrar Medicamentos:** es donde las enfermeras administran los medicamentos indicados por el medico.

4. **Realizar Estudios:** este rol se ejecuta cuando el paciente requiere de algún examen medico o de algún estudio para observar el estado en que se encuentra.

5. **Trasladar al Paciente:** es cuando el paciente es movido de la sala, puede ser trasladado a realizarse algún estudio, a piso cuando se encuentra estable pero aun necesita estar hospitalizado, a su casa si esta totalmente sano o a la morgue en caso de que fallezca.

6. **Actualizar Historia:** Día a día se registra el estado del paciente, los medicamentos administrados, los estudios realizados, de manera que se va creando la historia clínica del mismo.

Sala de Observación: Es la sala en donde se encuentran los pacientes que necesitan estar en observación o ameritan ser hospitalizado y se encuentran allí esperando la disponibilidad de una cama en piso. Posee los siguientes roles:

1. **Ingresar Pacientes:** Se encarga d recibir al paciente que viene de sala de curas para ser ingresado a la sala de observaciones, se le asigna una cama para ingresarlo.

2. **Observar al Paciente:** El mismo consiste en revisar cual es el diagnóstico que presenta el paciente, volver a examinarlo y darle los cuidados que sean necesarios.

3. **Administrar Medicamentos:** es donde las enfermeras administran los medicamentos indicados por el medico.

4. **Efectuar Estudios:** este rol se ejecuta cuando el paciente requiere de algún examen medico o de algún estudio para observar el estado en que se encuentra.

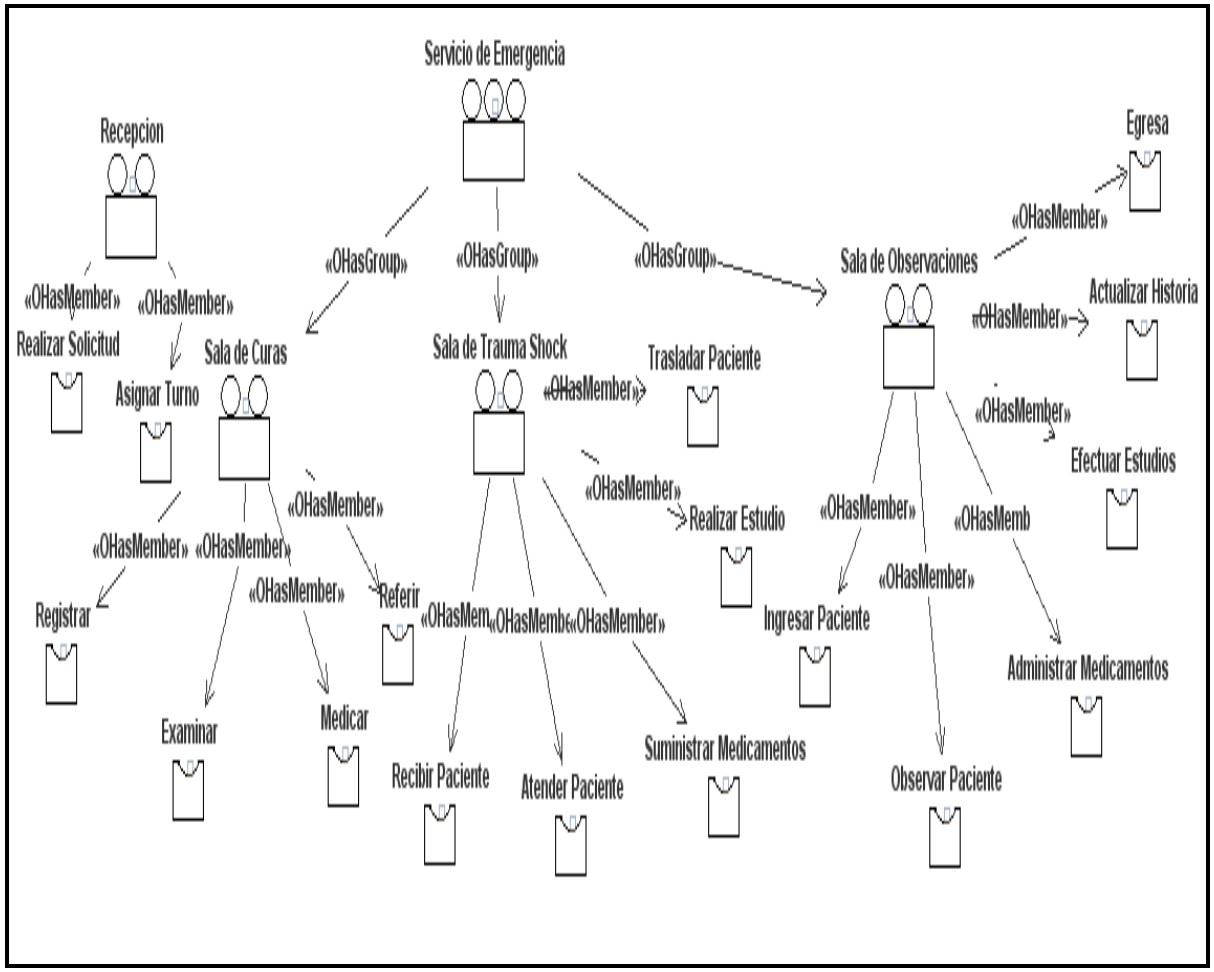
5. **Egresar al Paciente:** es cuando el paciente es movido de la sala, puede ser trasladado a realizarse algún estudio, a piso cuando se encuentra estable pero aun necesita estar hospitalizado, a su casa si esta totalmente sano o a la morgue en caso de que fallezca.

6. **Actualizar Historia:** Día a día se registra el estado del paciente, los medicamentos administrados, los estudios realizados, de manera que se va creando la historia clínica del mismo.

La siguiente figura muestra el modelo de la organización desarrollado con el editor IDK.

Figura 19

Modelo Organización



Fuente: Bieliukas (2010)

Modelo de Interacción:

El modelo de interacción describe el comportamiento que deben adoptar los agentes al recibir instrucciones de otro agente o de un humano, es decir, en el caso que se esta estudiando se muestra las relaciones e interacciones que existen entre los agentes, así como también los roles que se presentan.

A continuación se describe el diagrama de interacción para el servicio de emergencia del HCUAMP,

- **Solicitud en Ejecución:** consiste en ejecutar la solicitud del paciente, en donde se presenta una relación entre los agentes **Paciente- Recepcionista**, en donde el agente paciente realiza la solicitud para ser atendido y el recepcionista se encarga de registrar la solicitud.

- **Evaluación del Bachiller en Ejecución:** Consiste en describir el proceso que realiza el agente bachiller durante su desempeño en la sala de curas, aquí se presenta una relación entre los agentes **Paciente - Bachiller**, donde el agente paciente indica lo que siente y las causas por las cuales necesita ser atendido, el agente bachiller se encarga de examinar al paciente, luego registra la información del paciente.

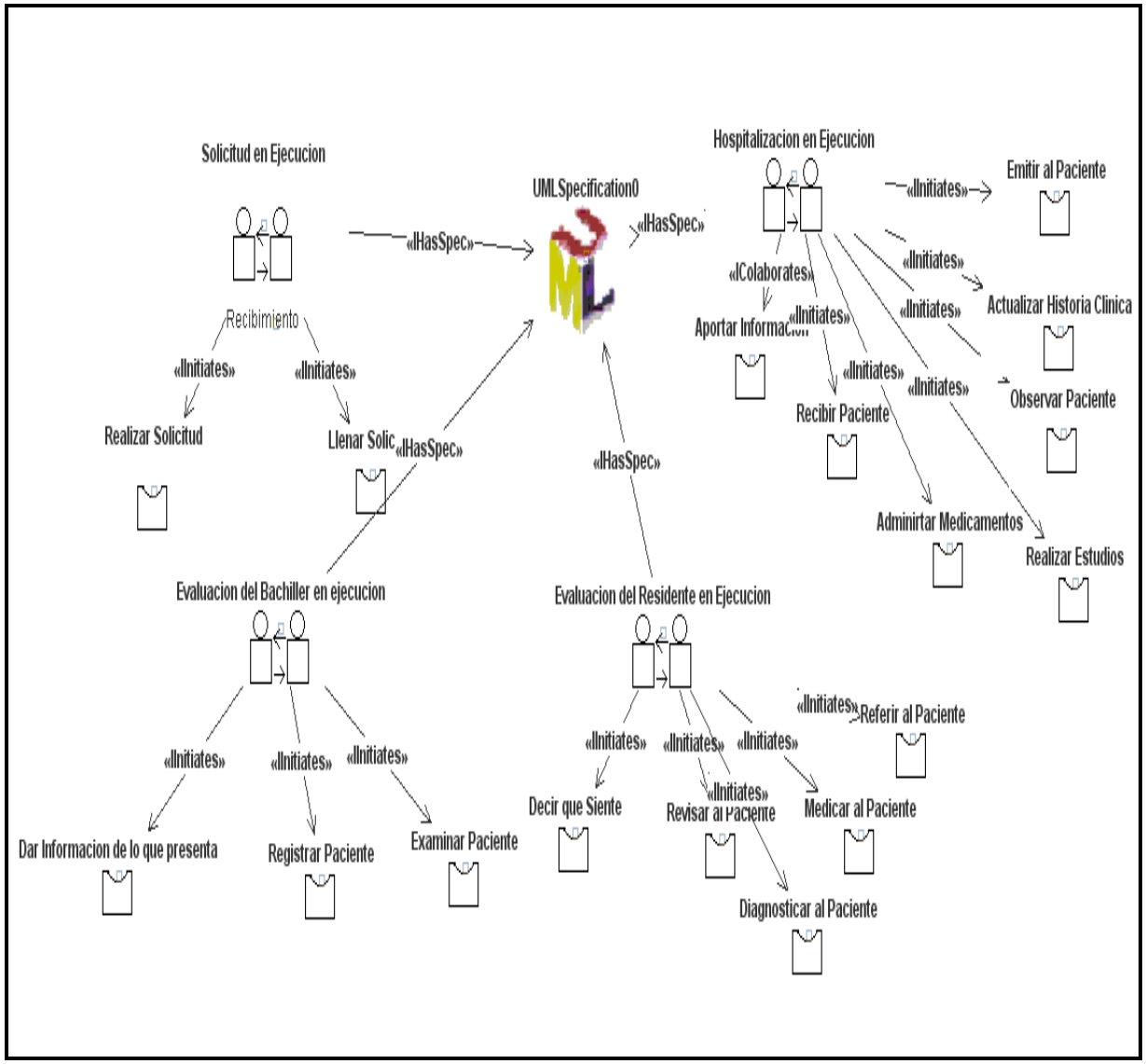
- **Evaluación del Residente en Ejecución:** Consiste en describir el proceso que realiza el agente Residente durante su desempeño en el servicio, aquí se presenta una relación entre los agentes **Paciente- Residente**, donde el agente paciente indica lo que siente y las causas por las cuales necesita ser atendido, el agente residente se encarga de examinar al paciente, diagnosticarlo, medicarlo y referirlo.

- **Hospitalización en Ejecución:** Consiste en describir el proceso que realiza el agente Emergenciologo durante su desempeño en el servicio, aquí se presenta una relación entre los agentes **Paciente- Emergenciologo**, donde el agente paciente aporta información de lo que siente, el agente emergenciologo se encarga de recibir , Examinar, Indicar administración de medicamentos, Indicar realización de estudios, observar , actualizar historia y emitir al paciente.

En la siguiente figura se muestra el modelo de interacción para la gestión de servicio de emergencia.

Figura 20

Modelo de Interacción



Fuente: Bieliukas (2010)

Cuadro 7

Entidades del modelo de interacción

Nombre	Tipo	Descripción
Solicitud en Ejecución	Interacción	modelo de interaccion:InteractionModel
UMLSpecification22	UMLSpecification	modelo de interaccion: InteractionModel
Realizar Solicitud	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Llenar Solicitud	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Evaluación del Bachiller en Ejecución	Interacción	modelo de interaccion: InteractionModel
Dar información de lo que presenta	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Revisar al paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Diagnosticar al paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Medicar al paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Referir al paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Hospitalizacion en ejecución	Interacción	modelo de interaccion: InteractionModel

Fuente: Bieliukas (2010)

Cuadro 8

Entidades del modelo de interacción (continuación)

Nombre	Tipo	Descripción
Aportar Información	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Administrar Medicamentos	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Realizar Estudios	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Observar Pacientes	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Actualizar Historia Clinica	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Emitir Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel

Fuente: Bieliukas (2010)

Cuadro 9

Relaciones que aparecen en el modelo de interacción

Tipo	Interacción	Rol
IInitates	Solicitud en Ejecución	Realizar Solicitud
IInitates	Solicitud en Ejecución	Llenar Solicitud
Haspec	Solicitud en Ejecución	UMLSpecification0
IInitates	Evaluación Bachiller en Ejecución	Dar Información de lo que presenta

Fuente: Bieliukas (2010)

Cuadro 10

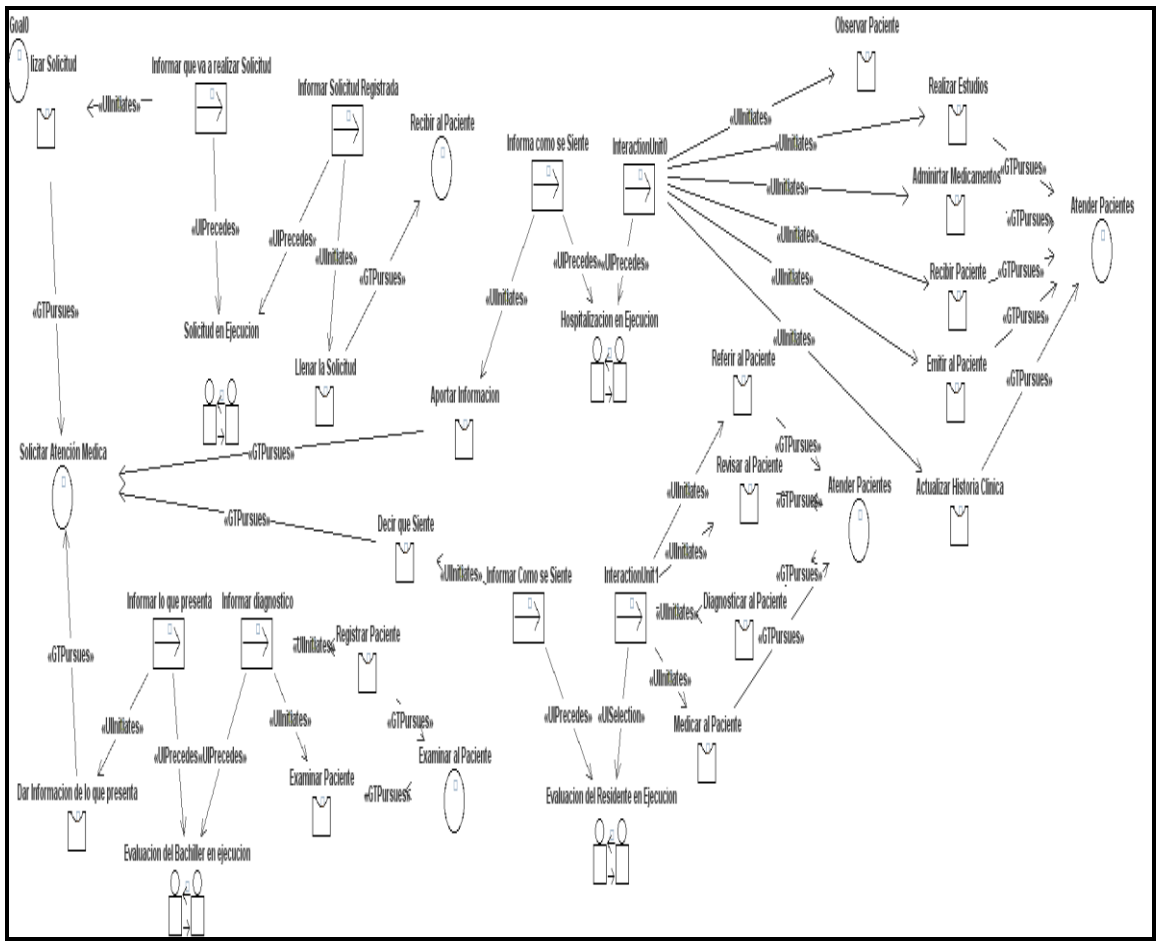
Relaciones que aparecen en el modelo de interacción (Continuación)

Tipo	Interacción	Rol
IInitates	Evaluación Bachiller en Ejecución	Registrar paciente
IInitates	Evaluación Bachiller en Ejecución	Examinar Paciente
Haspec	Evaluación Bachiller en Ejecución	UMLSpecification0
IInitates	Evaluación de Residentes en Ejecución	Decir que siente
IInitates	Evaluación de Residentes en Ejecución	Revisar al Paciente
IInitates	Evaluación de Residentes en Ejecución	Diagnosticar al Paciente
IInitates	Evaluación de Residentes en Ejecución	Medicar al Paciente
IInitates	Evaluación de Residentes en Ejecución	Referir al Paciente
Hapec	Evaluación de Residentes en Ejecución	UMLSpecification
IColaborates	Hospitalización en Ejecución	Aportar Información
IInitates	Hospitalización en Ejecución	Recibir al Paciente
IInitates	Hospitalización en Ejecución	Administrar Medicamentos
IInitates	Hospitalización en Ejecución	Realizar Estudios
IInitates	Hospitalización en Ejecución	Observar al Paciente
IInitates	Hospitalización en Ejecución	Actualizar Historia Clinica
IInitates	Hospitalización en Ejecución	Emitir Paciente
Hapec	Hospitalización en Ejecución	UMLSpecification0

Fuente: Bieliukas (2010)

Figura 21

Modelo de flujo Eventos



Fuente: Bieliukas (2010)

Cuadro 11

Entidades que aparecen en el modelo de flujo eventos

Nombre	Tipo	Descripción
Dar información de lo que presenta	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Examinar Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Decir que siente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Llenar Solicitud	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Registrar Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Aportar Información	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel

Fuente: Bieliukas (2010)

Cuadro 12

Entidades que aparecen en el modelo de flujo evento (continuación)

Nombre	Tipo	Descripción
Observar al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: interactionModel flujo de evento: InteractionModel
Realizar Estudios	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Administrar Medicamentos	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Pacientes	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Emitir al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Actualizar Historia Clínica	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Referir al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Revisar al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Diagnosticar al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Medicar al Paciente	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Realizar Solicitud	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Solicitar Atención Medica	Objetivo	Ojetivos y tareas: TasksAndGoalsModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Examinar al Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel

Fuente: Bieliukas (2010)

Cuadro 13

Entidades que aparecen en el modelo de flujo evento (continuación)

Nombre	Tipo	Descripción
Recibir al Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel modelo de interaccion:
Atender al Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel modelo de interaccion:
Realizar Estudios	Rol	Agent Rol asosiacion AgentModel modelo de interaccion: InteractionModel
Evaluación del Residente en Ejecución	Interacción	modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Evaluación del Bachiller en ejecución	Interacción	modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Solicitud en Ejecución	Interacción	modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Hospitalización en Ejecución	Interacción	modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Informar que va a realizar solicitud	Pase de Mensaje	flujo de evento: InteractionModel
Informar Solicitud Registrada	Pase de Mensaje	flujo de evento: InteractionModel
Informar lo que Presenta	Pase de Mensaje	flujo de evento: InteractionModel
Informar Diagnostico	Pase de Mensaje	flujo de evento: InteractionModel
Informar Como se Siente	Pase de Mensaje	flujo de evento: InteractionModel
Informar Estado Clínico del Paciente	Pase de Mensaje	flujo de evento: InteractionModel

Fuente: Bieliukas (2010)

Fase. 2 Elaboración:

Esta fase consiste en generar los modelos de agentes, con el fin de detallar los elementos de la arquitectura, así como también los modelos de tareas y objetivos para generar restricciones de control. En el diseño que corresponde a esta fase se expresa la ejecución de las tareas dentro de los modelos de interacción.

Modelo Agente y Roles:

Este modelo describe el funcionamiento de cada uno de los agentes del sistema, allí se reflejan los roles que juegan. Los agentes que pertenecen al prototipo del sistema multiagente para la gestión de servicios de emergencia médica hospitalaria son los siguientes:

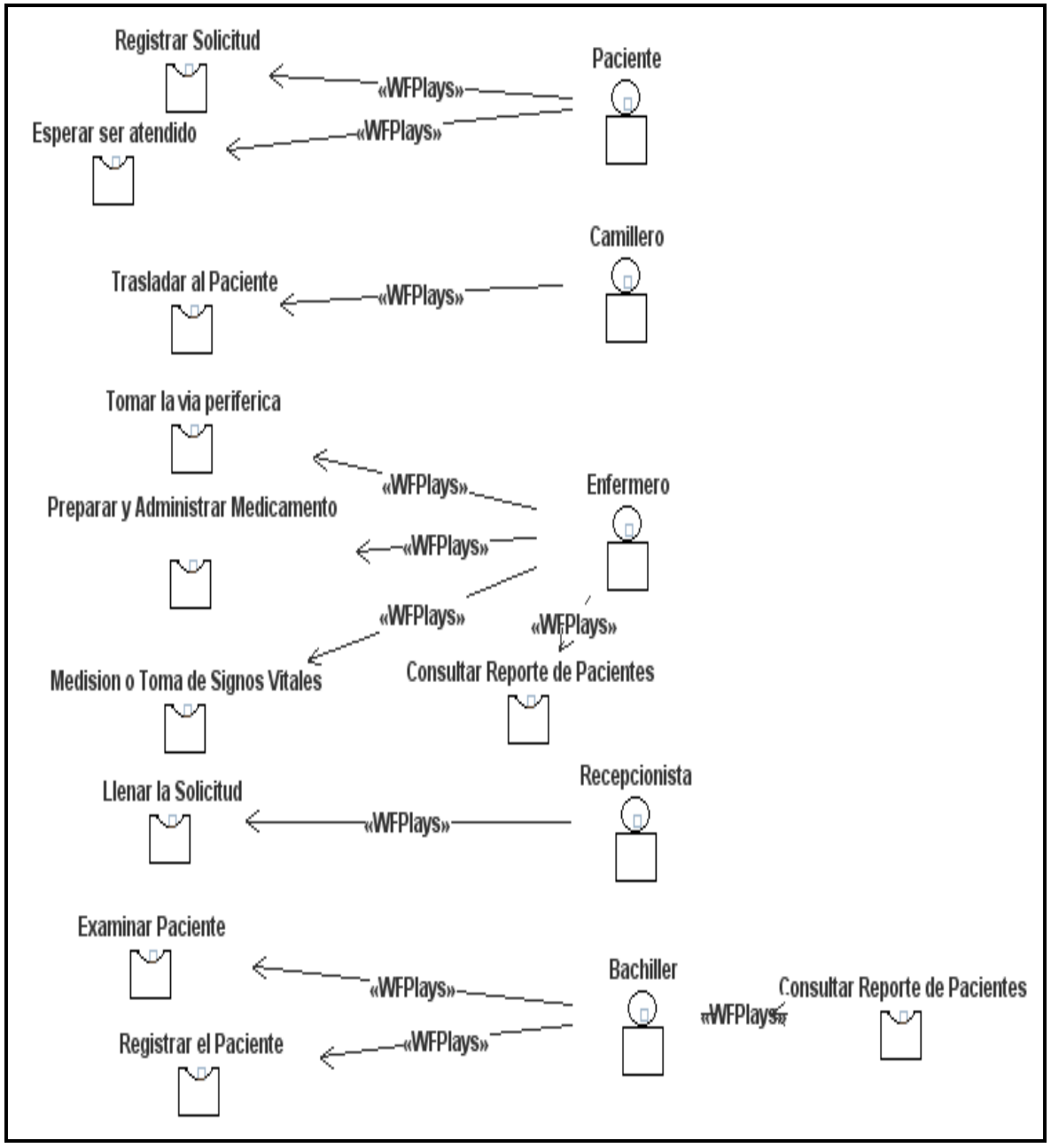
- **Agente Paciente:** Este agente es el que se encarga de realizar el papel del paciente que solicita el servicio de emergencia, el mismo se encargara de realizar la solicitud para ser atendido y posteriormente debe esperar que venga su turno para ser atendido.
- **Camillero:** Son los que se encargan de trasladar al paciente al sitio que indica el medico. A continuación se muestra el diagrama con la descripción de este agente.
- **Enfermero:** Son los que se encargan de preparar y administrar los medicamentos indicados por el medico (residente, emergenciologo, especialista), así como también estar pendiente de los signos vitales del paciente, tomar las vías periféricas.
- **Bachiller:** Son los estudiantes de medicina, que se encargan de atender al paciente en lo que ingresa, le brindan los primeros auxilios en la sala de cura y examina al paciente junto con los residentes, también se encargan de registrar al paciente, en cuanto a sus datos, diagnostico y lugar a donde fue referido.

- **Residente:** Son los estudiantes del posgrado de emergencia, ellos se encargan de estar pendiente de todos los pacientes que se encuentran en el servicio de emergencia, guiados por los especialistas de emergencia. Por otra parte se encarga de actualizar la historia clínica del paciente.
- **Emergenciologo:** Son los médicos especialistas en emergencia, ellos se encargan del cuidado de los pacientes de una manera mas detallada que los residentes, indican medicamentos, estudios y son los encargados de comunicarse con los distintos tipos de especialista si existe algún caso particular. A su vez se encarga de chequear la historia clínica del paciente y de actualizarla. Por otra parte también es el que autoriza el traslado del paciente a piso si va ser hospitalizado, a su casa si se le va a dar de alta o a la morgue en caso de fallezca.
- **Consulta Interno:** Este agente es utilizado por los profesionales de la medicina para tener información sobre el paciente.
- **Consulta externa:** Es el que se encarga de brindarle información a los familiares del paciente en cuanto a su estado y lugar donde se encuentra.
- **Recepcionista:** Es el que se encarga de realizarle al paciente la solicitud para ser atendido, colocando el número de cédula en el sistema y registrando las causas por la que realiza la solicitud. Posteriormente cuando reciba respuesta del turno que se le asignó al paciente, este se encarga de darle la información al paciente.
- **Cola:** Es el que se encarga de generar la prioridad de la solicitud registrada, de acuerdo a criterios definidos previamente.
- **Familia:** E**s** el familiar del paciente que desea saber donde se encuentra el paciente que esta acompañando.

A continuación se muestra el diagrama del modelo de agente

Figura 22

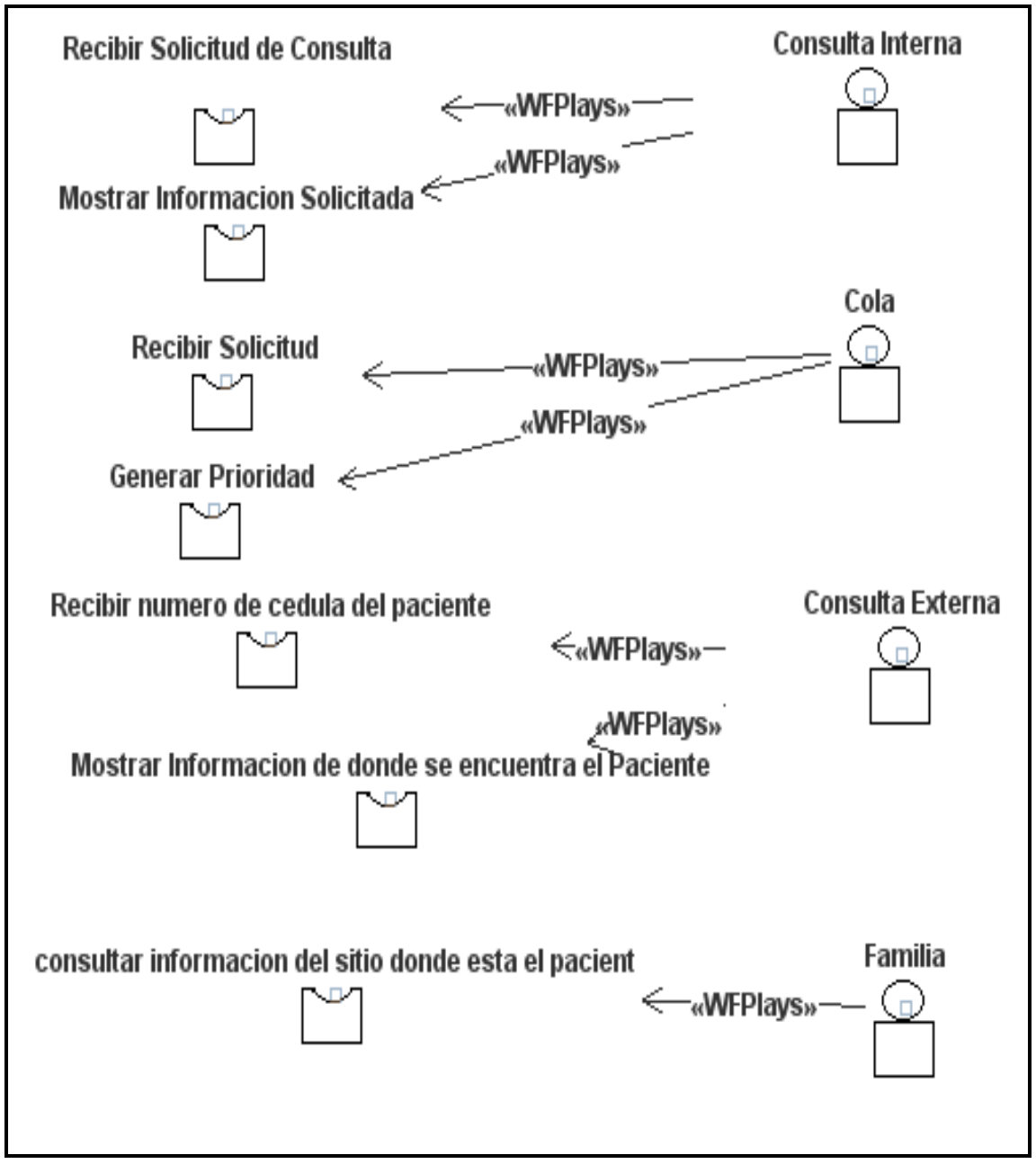
Modelo de Agentes y Roles



Fuente: Bieliukas (2010)

Figura 23

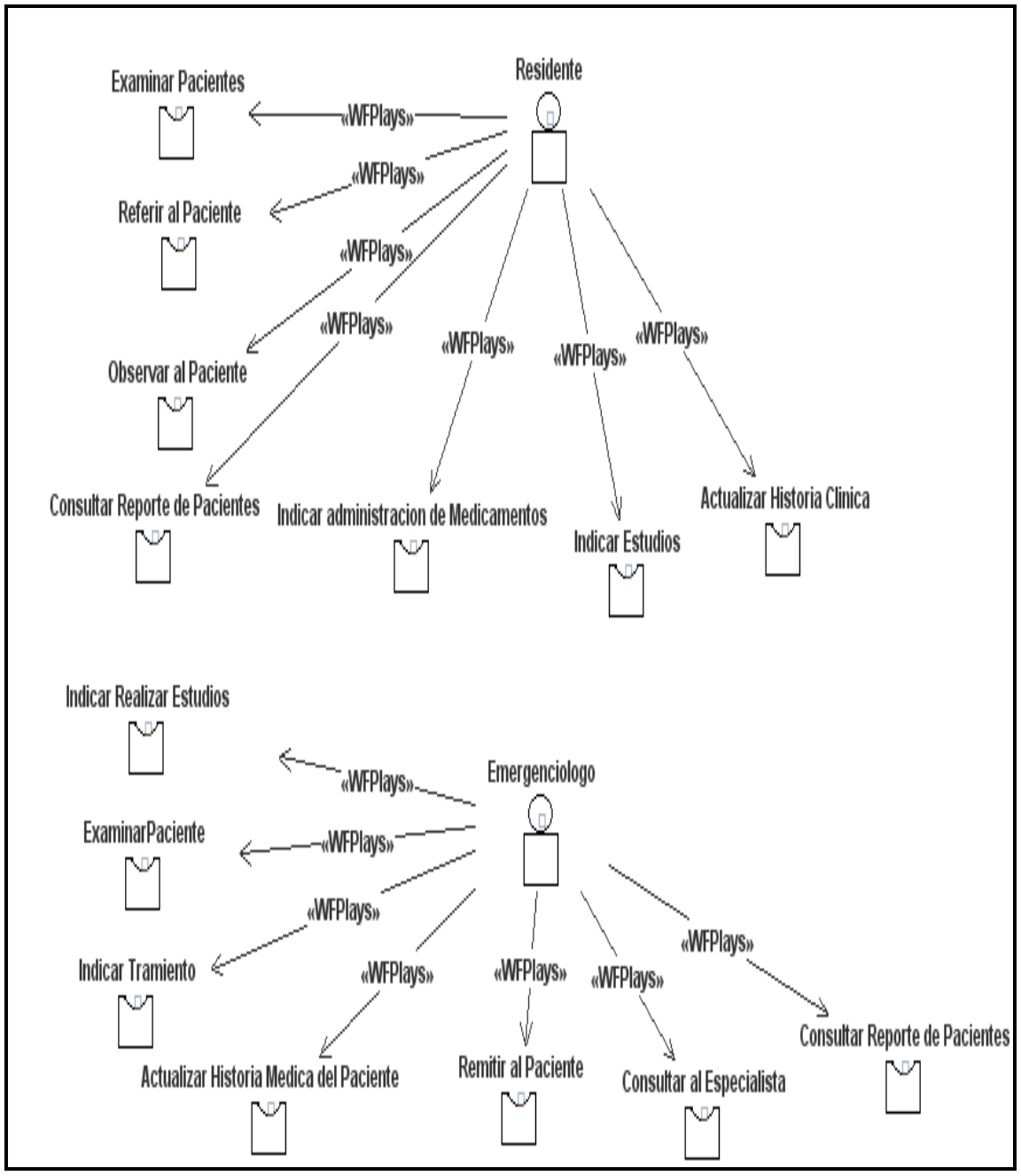
Modelo de Agentes y Roles (continuación)



Fuente: Bieliukas (2010)

Figura 24

Modelo de Agentes y Roles (continuación)



Fuente: Bieliukas (2010)

Cuadro 14

Entidades que aparecen en el Modelo de Agentes y Roles

Nombre	Tipo	Descripción
Paciente	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Camillero	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Bachiller	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Residente	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Emergenciologo	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Enfermera	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Especialista	Agente	Agent Rol asosiacion: AgentModel Objetivos y tareas: TasksAndGoalsModel
Consulta Interna	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Consulta Externa	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Cola	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Recepcionista	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Familia	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel

Fuente: Bieliukas (2010)

Cuadro 15

Entidades que aparecen en el Modelo de Agentes y Roles (continuación)

Nombre	Tipo	Descripción
Registrar Solicitud	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Esperar ser Atendido	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Trasladar al Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Tomar via Periferica	Rol	Agent Rol asosiacion: AgentModel modelo de interaccionInteractionModel flujo de evento: InteractionModel
Preparar y Administrar Medicamento	Rol	Agent Rol asosiacion: AgentModelÇmodelo de interaccion: InteractionModel flujo de evento: InteractionModel
Medir o Tomar Signos Vitales	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Llenar Solicitud	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Examinar Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Registrar Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Referir al Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Observar al Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Consultar del sitio donde se encuentra el paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel

Fuente: Bieliukas(2010)

Cuadro 16

Entidades que aparecen en el Modelo de Agentes y Roles (continuación)

Nombre	Tipo	Descripción
Indicar Administración de Medicamentos	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Indicar Estudios	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Actualizar Historia Medica	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Remitir al Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Consultar Especialista	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Solicitud de Consulta	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Mostrar Información Solicitada	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Solicitud	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Generar Prioridad	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Numero de Cedula del Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Mostrar Información donde se encuentra el Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Consultar Informacion del Paciente	Rol	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel

Fuente: Bieliukas (2010)

Cuadro 17

Relaciones que aparecen en el Modelo de Agentes y Roles

Tipo Relación	Agente	Rol
WFPlays	Paciente	Registrar Solicitud
WFPlays	Paciente	Esperar ser atendido
WFPlays	Camillero	Trasladar Paciente
WFPlays	Enfermero	Tomar Vía Periférica
WFPlays	Enfermero	Preparar y Administrar Medicamento
WFPlays	Recepcionista	Llenar la Solicitud
WFPlays	Bachiller	Examinar Paciente
WFPlays	Bachiller	Registrar Paciente
WFPlays	Residente	Examinar Paciente
WFPlays	Residente	Referir Paciente
WFPlays	Residente	Observar Paciente
WFPlays	Residente	Indicar Administración de Medicamentos
WFPlays	Residente	Actualizar Historia Clínica
WFPlays	Emergenciólogo	Indicar Realizar Estudios
WFPlays	Emergenciólogo	Indicar Medicamento
WFPlays	Emergenciólogo	Examinar Paciente
WFPlays	Emergenciólogo	Actualizar Historia del Paciente
WFPlays	Emergenciólogo	Remitir al Paciente
WFPlays	Emergenciólogo	Consultar Especialista
WFPlays	Consulta Interna	Recibir Solicitud de Consulta
WFPlays	Consulta Interna	Mostrar Información Solicitada
WFPlays	Cola	Recibir Solicitud
WFPlays	Cola	Generar Prioridad
WFPlays	Consulta Externa	Recibir Numero de Cedula del Paciente
WFPlays	Consulta Externa	Mostrar Información del Sitio donde se
WFPlays	Familia	Consultar Información del Sitio donde

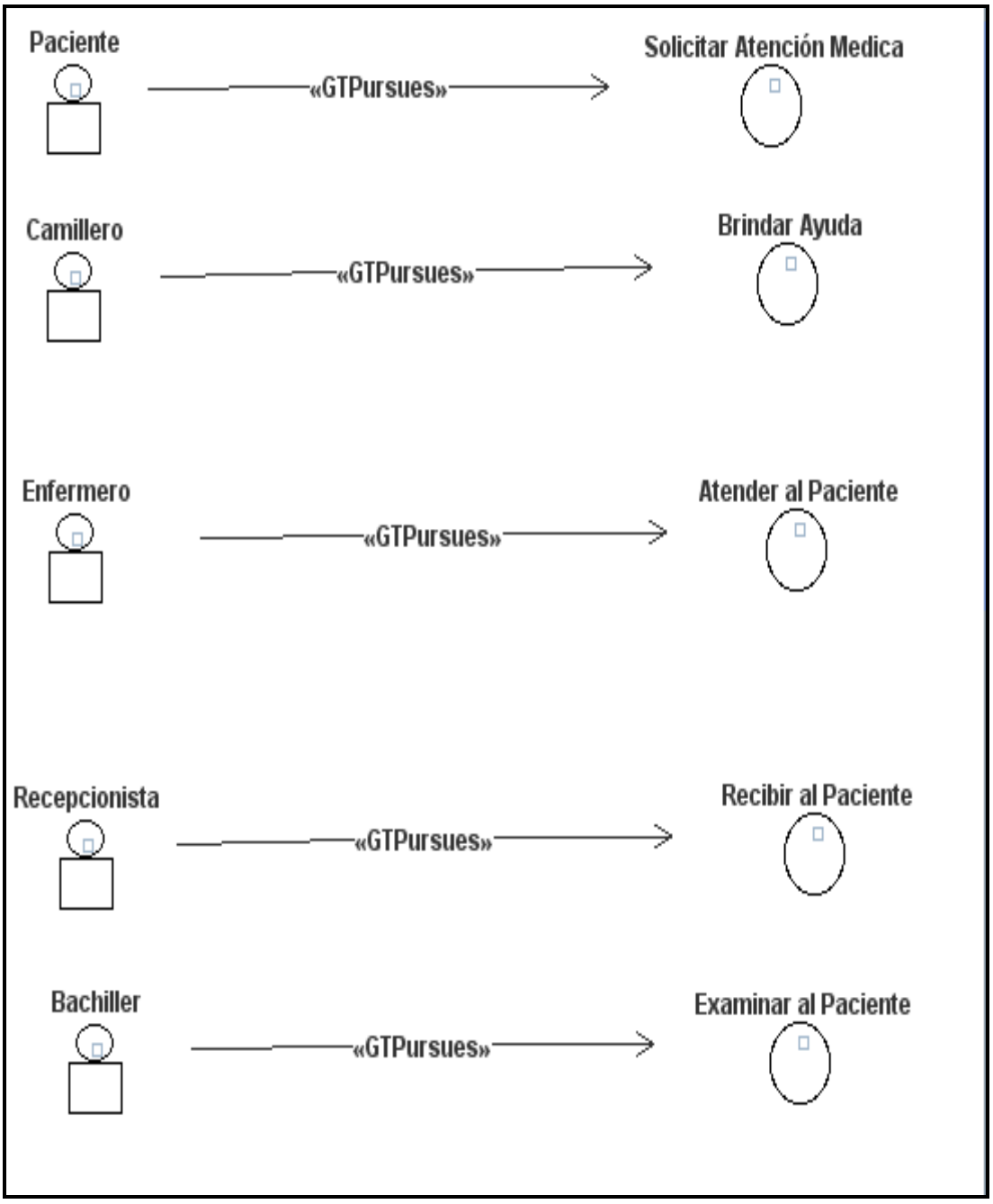
Fuente: Bieliukas (2010)

Modelo de Objetivos y Tareas:

Este modelo consiste en describir el objetivo o meta donde quieren llegar los agentes. A continuación se especifica en los diagramas los objetivos que poseen los agentes del sistema de gestión de servicio.

Figura 24

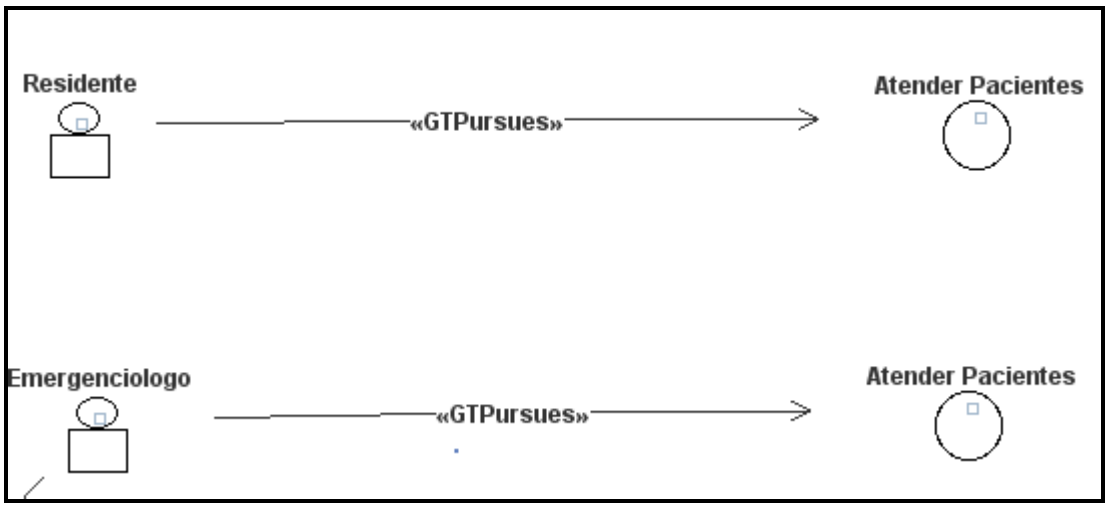
Modelo de Objetivos y Tareas



Fuente: Bieliukas (2010)

Figura 25

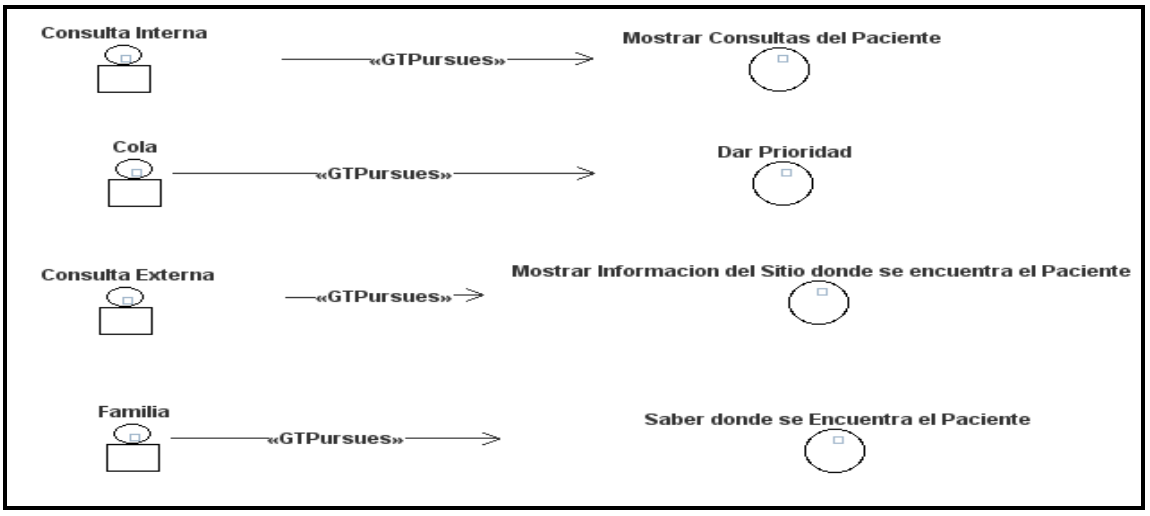
Modelo de Objetivos y Tareas (Continuación)



Fuente: Bieliukas (2010)

Figura 26

Modelo de Objetivos y Tareas (Continuación)



Fuente: Bieliukas (2010)

Cuadro 18

Entidades que aparecen en el Modelo de Objetivos y Tareas

Nombre	Tipo	Descripción
Solicitar Atención Medica	Objetivo	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Brindar Ayuda	Objetivo	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Atender Paciente	Objetivo	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Recibir Paciente	Objetivo	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Examinar Paciente	Objetivo	Agent Rol asosiacion: AgentModel modelo de interaccion: InteractionModel flujo de evento: InteractionModel
Mostrar Consulta del Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel
Dar Prioridad	Objetivo	Ojetivos y tareas: TasksAndGoalsModel
Mostrar Información del Sitio donde se encuentra el Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel
Saber donde se Encuentra el Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel
Examinar Paciente	Objetivo	Ojetivos y tareas: TasksAndGoalsModel
Paciente	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Camillero	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Bachiller	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Residente	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Emergenciólogo	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Enfermero	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel

Fuente: Bieliukas (2010)

Cuadro 19

Entidades que aparecen en el Modelo de Objetivos y Tareas (Continuación)

Nombre	Tipo	Descripción
Cola	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Consulta Interna	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Consulta Externa	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Familia	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel
Recepcionista	Agente	Agent Rol asosiacion: AgentModel Ojetivos y tareas: TasksAndGoalsModel

Fuente: Bieliukas (2009)

Cuadro 20

Relaciones que aparecen en el Modelo de Objetivos y Tareas

Tipo Relación	Agente	Rol
GTPursues	Paciente	Solicitar Atención Médica
GTPursues	Camillero	Brindar Ayuda
GTPursues	Enfermero	Atender Paciente
GTPursues	Recepcionista	Recibir al Paciente
GTPursues	Bachiller	Examinar Paciente
GTPursues	Residente	Atender Paciente
GTPursues	Emergenciólogo	Atender Paciente
GTPursues	Consulta Interna	Mostrar Consultas del Paciente
GTPursues	Consulta Externa	Mostrar Información del Sitio donde se encuentra el Paciente
GTPursues	Familia	Saber donde se Encuentra el Paciente
GTPursues	Cola	Dar Prioridad

Fuente: Bieliukas (2010)

Fase 3. Construcción:

Esta fase consiste en verificar detalladamente los modelos diseñados, ya culminado todos los modelos, se procede a generar el código que se obtiene a través de la herramienta IDK. En el anexo A se encuentra el código fuente generado y en el Anexo B se muestra la corrida que se obtuvo al ejecutar el código fuente.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

Los servicios de emergencia son un pilar fundamental en el área de salud, debido a que es el lugar donde se atienden a todas las personas que requieren atención médica inmediata. Los sistemas multiagentes proporcionan una herramienta para controlar el flujo de procesos necesarios para administrar la atención de los pacientes que ingresan a los servicios de emergencias.

El presente trabajo de investigación proporciona una serie de conclusiones basadas en las experiencias vividas durante el desarrollo del mismo. Entre las cuales se pueden mencionar:

- El principal aporte de esta investigación, es que se logró simular el funcionamiento de todos los procesos que ocurren dentro del servicio y en particular se logró optimizar el proceso de recepción de los pacientes, que es uno de los procesos que presenta en la actualidad mayor dificultad

- Se logró desarrollar de forma satisfactoria los requerimientos planteados por parte del servicio de emergencia del Hospital Central Universitario "Antonio María Pineda".

- Se cumplió con los objetivos específicos establecidos en la investigación.

- El prototipo del sistema multiagente proporciona información específica y detallada sobre la atención de los pacientes que solicitan atención en el servicio.

- Se detectaron diversas ventajas del prototipo del sistema frente a la situación actual que presenta el servicio de emergencia, se mejora notablemente la recepción de los pacientes en el servicio, debido a que todos serán atendidos, por la prioridad que presentan y a su vez los pacientes serán informados constantemente cuanto tiempo les queda para ser atendidos.

- Con la aplicación de esta propuesta es posible, realizar un seguimiento a cada departamento para controlar, la calidad de atención prestada del paciente, con la aplicación de esta propuesta todos los pasos realizados por cada departamento quedaran reflejados en el sistema.

- La solicitud que realiza el paciente para ser atendido de forma automática y el turno que se le asigna es basándose en determinados parámetros dependiendo de la prioridad que presenta siendo el SMA en este sentido versátil y adaptable.

Es oportuno mencionar que cumplido con los objetivos planteados en la investigación y concluido con el prototipo del sistema multiagente, nos queda plantear algunas recomendaciones que ayuden a afinar y mejorar la gestión del servicio de emergencia. Entre las cuales se pueden mencionar:

- Desarrollar el Sistema tomando como base el prototipo del sistema multiagente para que efectivamente se le pueda dar uso en el servicio de emergencia.

- Invertir en equipos de computación para la implementación del sistema.

- Crear normativas en relación al funcionamiento del sistema para que funcione de manera correcta.

- Debe tomarse en cuenta que es el primer paso hacia la futura automatización del proceso de atención al paciente en el servicio de

emergencia y como tal, es necesario el estudio de todas las condiciones que van a afectar su comportamiento.

- La elección de un sistema basado en agentes la modificación o actualización sin detener la ejecución, debido a que los agentes ejercen su capacidad de adaptación y son capaces de reorganizarse.

REFERENCIAS BIBLIOGRÁFICAS

Botía y Otros. 2008. **"The INGENIAS Project: Methods And Tool For Developing Multiagent" Systems.** IEEE LATIN AMERICA TRANSACTIONS**.**

Bratman 1987. **"Intentions, Plans, and Practical Reason".** Libro completo. Harvard University Press.

Bravo 2003. **Propuesta de un Sistema de Gestión de Servicios Multiagentes para el SCDIA**. Mérida: Universidad de Los Andes, Facultad de Ingeniería, Postgrado de Computación.

Caire y Otros. 2002. **"Agent Oriented Analysis using MESSAGE/UML"**. Actas de conferencia. Springer Verlag.

Cázares y Otros. 2000. **"Investigación de Campo e Investigación Documental".**

Claret 2005. **"¿Como hacer y defender una tesis?".** Editorial Texto, C.A. Cuarta Edición.

Clemmer y Gardem . 1995. **"Informática Médica en la unidad de cuidados intensivos".**

Collis y Ndumu. 1999. **"The Role Modelling Guide. Informe. Applied Research and Technology"**.

Cuesta. 2005, **Ingeniería de Software orientada a agente.** España: Universidad de Vigo, departamento de Informática. Grupo Web de agentes inteligentes.

DeLoach. 2001. **"Analysis and Design using MaSE and agentTool".** Actas de conferencia. Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conferece (MAICS).

Gallego y otros 2004. **"Breve análisis de algunas metodologías de Diseño SMA"**. España: Universidad de Alicante, Departamento de Ciencias de la Computación e Inteligencia Artificial.

García. 2006**. "Modelando el Proceso de Desarrollo de INGENIAS con EMF".** Departamento de Ingeniería del Software e Inteligencia Artificial. Universidad Complutense de Madrid (SPAIN) .Universidad de Vigo. Ed. Politécnico.

Gómez y Pavón. 2004. "**Desarrollo de Sistemas Multi-Agente con INGENIAS**" Departamento de Sistemas Informáticos y Programación de la Universidad Complutense Madrid.

Gómez. 2002**. Modelado de Sistemas Multi-Agente.** Tesis Doctoral, Universidad Complutense de Madrid.

Gómez. 2003. **Metodología para el Desarrollo de Sistemas Multi – agentes**, Madrid: Departamento de Sistemas Informáticos y Programación. Facultad de Informática, Universidad Complutense.

Iglesias. 1998. **"Definición de una metodología para el desarrollo de Sistemas Multi-Agente. Tesis doctoral"**. Departamento de ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid.

Informe del movimiento hospitalario del Hospital Central Universitario "Antonio María Pineda".

"INGENIAS". 2005. Universidad Complutense de Madrid (GRASIA). http://grasia.fdi.ucm.es/ingenias. (Consulta Agosto 2009).

Jennings. 1998. **"Applications of Intelligent Agents. Queen Mary & Westfield College"**. University of London**.**

Kinny. y Georgeff. 1997. **"Modelling and Design of Multi-Agent Systems, Springer Verlag"**.

La Dirección Nacional de Servicios Académicos Virtuales de la Universidad Nacional de Colombia http://www.virtual.unal.edu.co/cursos/ingenieria/2001394/docs_curso/capitulo1/leccion1.2.htm (Consulta Agosto 2009)

Manual para presentación del trabajo de grado académico de Especialización, Maestría y doctorado de la UCLA. 2002.

Mas 2005**. "Agentes de Software y Sistemas Multiagentes"**. Pearson – Prentice Hall.

Pérez. 2007. **"Aplicación de Metodologías INGENIAS, ZEUS, MASINA al desarrollo de Sistemas Multi-Agente, partiendo de SMA de Subastas para la identificación de Mejores practicas"** Departamento de ingeniería eléctrica, electrónica, sistemas y telecomunicaciones, de la facultad de ingenierías y arquitectura. Universidad de Pamplona Colombia.

Portal de la Gobernación del Estado Aragua. URL: http://www.aragua.gob.ve (Consulta Abril 2009).

Pressman. 1982. **"Software Engineering: A Practitioner's Approach"**. Libro completo. McGraw-Hill Series in Software Engineering and Technology. McGraw-Hill.

Programa avanzado de apoyo vital en trauma para médicos 2005.

Russell y Norvig. 1996**, Inteligencia Artificial. Un Enfoque Moderno,** Prentice Hal.

Salvador. 1996. **“Sistema de información cmbd como herramienta de gestión y control de calidad asistencial”.** Departamento de microbiología, medicina preventiva y salud publica facultad de medicina de la universidad de Zaragoza España.

Serrano y Suros presentaron un “**Diseño de un Sistema Multiagente para la Gestión de Recursos usando la Metodología MAS CommonKADS”**. Caracas: Universidad Central de Venezuela, facultad de ciencias. Centro de Computación Paralela y Distribuida.

Sociedad Venezolana de Medicina de Emergencia y Desastre, en la presentación de las **“Bases Legales de la Atención Prehospitalaria en Venezuela”**.

Tansley y Haybal. 1993. **“Knowledge Based systems Analysis and Design a KADS developer's handbook”**. Libro completo. Prentice Hall.

Villatoro. 2005. **“DEFINICIONES BÁSICAS EN MEDICINA DE URGENCIA”.** Medico adscrito al servicio de urgencias American British Cowdray Campus Santa Fe, UMAE General CMN Raza IMSS Profesor Titular Urgencias Medico Quirúrgicas ESM IPN.

Weiss 1998. **Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence**. MIT Press, Cambridge, MA, USA

Wooldridg. 1998. **“Agent Technology Foundations and Markets”. edicion G.Weiss.**

Wooldridg. 2000 **“An Introducction to Multiagent Systems”. edicion G.Weiss.**

Wooldridg 1995. **“Multiagent Systems” edicion G.Weiss.**

Zambonelly y Otros. 2000. **“Organisational Rules as an Abstraction for the Analysis and Design of Multi-Agent Systems”.** International Journal of Software Engineering and knowledge Engineering.

ANEXOS

ANEXO A

CÓDIGO FUENTE

Código Fuente generado por el editor IDK

/* Copyright (C) 2005 Jorge Gomez Sanz

This file is part of INGENIAS Agent Framework, an agent infrastructure linked to the INGENIAS Development Kit, and availabe at http://grasia.fdi.ucm.es/ingenias or http://ingenias.sourceforge.net.

INGENIAS Agent Framework is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

INGENIAS Agent Framework is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with INGENIAS Agent Framework; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

*/

package ingenias.jade;
import java.io.File;
import java.io.FileInputStream;

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import jade.core.*;
import ingenias.jade.mental.*;
import ingenias.jade.graphics.MainInteractionManager;
public class MainThreeAgentsDeploymentProdStandAlone {

  public static void main(String args[]) throws Exception{

        IAFProperties.setGraphicsOn(false);

              new Thread(){

                    public void run(){

                          String[] args1=new String[4];
                          args1[0]="-port";
                          args1[1]="60000";
                          args1[2]="-file-dir";
                          args1[3]="jade/";

                          jade.Boot.main(args1);

                    }

              }.start();

      // Get a hold on JADE runtime

      jade.core.Runtime rt = jade.core.Runtime.instance();

      // Exit the JVM when there are no more containers around

      rt.setCloseVM(true);

      // Create a default profile

      Profile p = new ProfileImpl();
```

```
p.setParameter("preload","a*");
p.setParameter(Profile.MAIN_PORT, "60000");
p.setParameter(Profile.FILE_DIR, "jade/");
 // Waits for JADE to start
boolean notConnected=true;
        while (notConnected){
                try {
                        Socket s=new
Socket("localhost",Integer.parseInt("60000"));
                        notConnected=false;
                } catch (NumberFormatException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (UnknownHostException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (IOException e) {
                                System.err.println("Error:
"+e.getMessage());
                        System.err.println("Reconnecting in one
second");
                        try {
                                Thread.currentThread().sleep(1000);
                        } catch (InterruptedException e1) {
                                // TODO Auto-generated catch block
                                e1.printStackTrace();
                        }
                }
        }
// Create a new non-main container, connecting to the default
```

```
        // main container (i.e. on this host, port 1099)
        final jade.wrapper.AgentContainer ac = rt.createAgentContainer(p);
{
        // Create a new agent
        final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Paciente",
           "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
            new Thread(){
          public void run(){
           try {
              System.out.println("Paciente");
             Paciente.start();
           } catch (Exception e){
            e.printStackTrace();
           }
          }
        }.start();
        // Create a new agent
        final jade.wrapper.AgentController  Paciente=
ac.createNewAgent("Recepcionista",
           "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
            new Thread(){
          public void run(){
           try {
              System.out.println("Recepcionista");
             Recepcionista.start();
           } catch (Exception e){
            e.printStackTrace();
           }
```

```
  }
}.start();
// Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Enfermero",
   "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
    new Thread(){
 public void run(){
  try {
     System.out.println("Enfermero");
    Enfermero.start();
  } catch (Exception e){
    e.printStackTrace();
  }
 }
}.start();
// Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Bachiller",
   "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
    new Thread(){
 public void run(){
  try {
     System.out.println("Bachiller");
    Bachiller.start();
  } catch (Exception e){
    e.printStackTrace();
  }
 }
```

```
    }.start();
   // Create a new agent
    final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Residente",
      "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
       new Thread(){
     public void run(){
      try {
        System.out.println("Residente");
       Residente.start();
      } catch (Exception e){
       e.printStackTrace();
      }
     }
    }.start();
      // Create a new agent
    final jade.wrapper.AgentController  Paciente=
ac.createNewAgent("Emergenciologo",
      "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
       new Thread(){
     public void run(){
      try {
        System.out.println("Emergenciologo");
       Emergenciologo.start();
      } catch (Exception e){
       e.printStackTrace();
      }
     }
    }.start();
```

```
// Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Familia",
    "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
     new Thread(){
   public void run(){
    try {
      System.out.println("Familia");
     Familia.start();
    } catch (Exception e){
     e.printStackTrace();
    }
   }
  }.start();
  // Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Consulta
Interna",
    "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
     new Thread(){
   public void run(){
    try {
      System.out.println("Consulta Interna ");
     ConsultaInterna.start();
    } catch (Exception e){
     e.printStackTrace();
    }
   }
  }.start();
  // Create a new agent
```

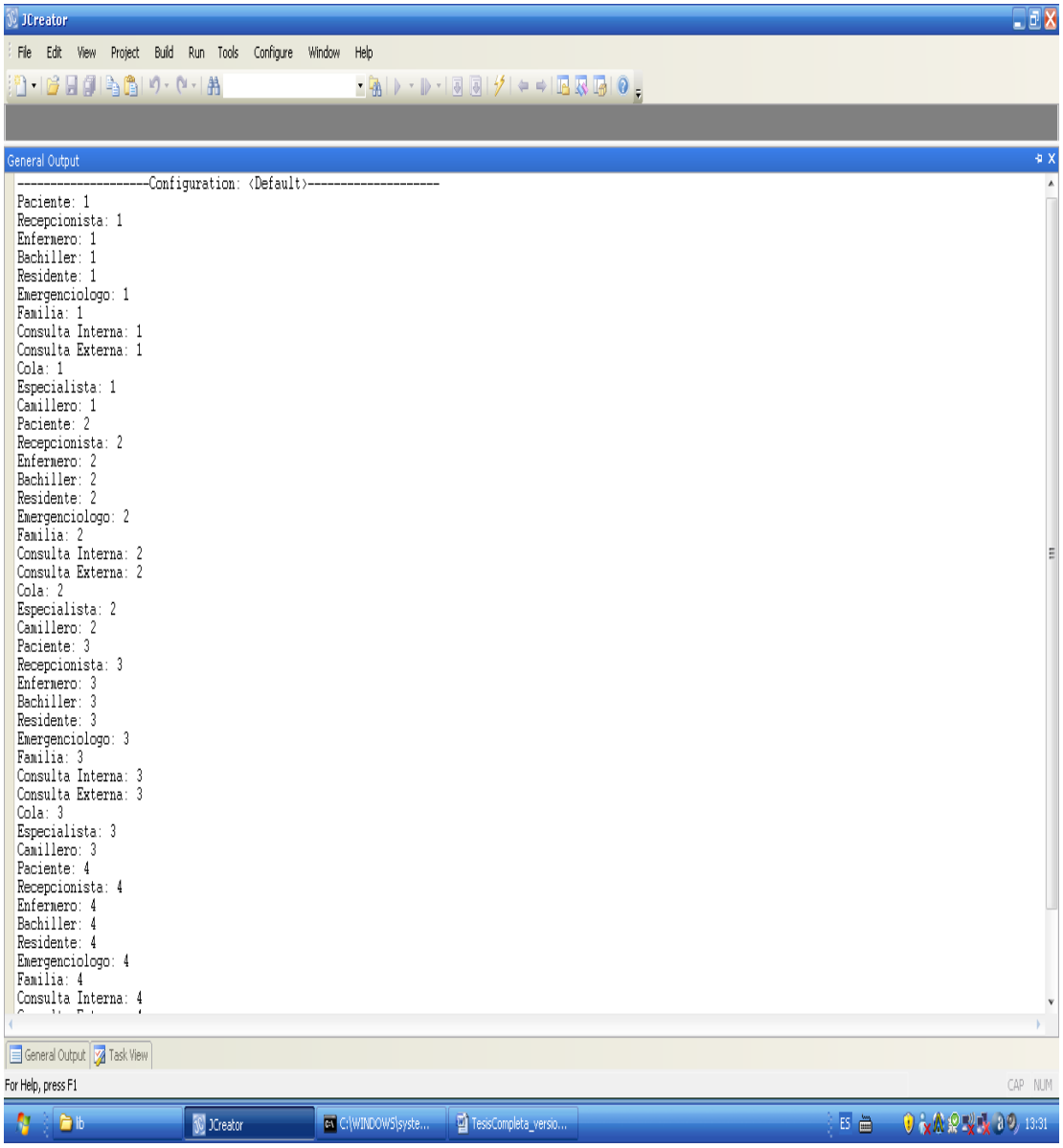
```
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Consulta
Externa",
          "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
          new Thread(){
        public void run(){
          try {
            System.out.println("Consulta  Externa");
           ConsultaExterna.start();
          } catch (Exception e){
           e.printStackTrace();
          }
        }
       }.start();
// Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Cola",
          "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);
          new Thread(){
        public void run(){
          try {
            System.out.println("Cola");
           Cola.start();
          } catch (Exception e){
           e.printStackTrace();
          }
        }
       }.start();
// Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Especialista",
```

```
        "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);

        new Thread(){
      public void run(){
        try {
          System.out.println("Especialista");
         Especialista.start();
        } catch (Exception e){
         e.printStackTrace();
        }
      }

    }.start();
    // Create a new agent
final jade.wrapper.AgentController  Paciente= ac.createNewAgent("Camillero",

        "ingenias.jade.agents.HelloWorldAgentJADEAgent", new Object[0]);

        new Thread(){
      public void run(){
        try {
          System.out.println("Camillero");
         Camillero.start();
        } catch (Exception e){
         e.printStackTrace();
        }
      }

    }.start();


            MainInteractionManager.getInstance().setTitle("node
ThreeAgentsDeployment");

  }
```

ANEXO B

CORRIDA DEL PROTOTIPO

Corrida del Prototipo



Fuente: Bieliukas (2010)