INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA

MAESTRÍA EN SISTEMAS COMPUTACIONALES

“Monitoreo y tutor de plan alimenticio para regular pacientes con diabetes mellitus tipo 2”.

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN SISTEMAS COMPUTACIONALES

P R E S E N T A

OMAR LAGUNES DURAN

ASESOR: DR. LUIS ALBERTO MORALES ROSALES

MISANTLA, VERACRUZ MAYO, 2015

RESUMEN

En esta tesis, la investigación se enfoca en el diseño e implementación de un sistema multiagente, para proporcionar una recomendación alimenticia adecuada a un paciente con diabetes mellitus tipo 2 tomando en cuenta sus preferencias alimenticias tanto como su glucosa, presión arterial y actividad física.

Este trabajo en particular, se centra en resolver los problemas de control de la glucosa y la presión arterial en un paciente con Diabetes Mellitus Tipo 2 (DM2) mediante una recomendación alimenticia. También proporciona una herramienta para resolver el problema de la gestión de grandes cantidades de información provenientes de su automonitoreo de glucosa y presión arterial. Por otro lado genera alertas en el caso que alguno de estos datos de automonitoreo esté en un rango peligroso para su salud.

El sistema está formado por tres agentes: Agente Interfaz, Agente Dieta y Agente Evaluación. La comunicación entre estos y cada una de sus tareas específicas ayudan al usuario al control y monitoreo de la diabetes mellitus tipo 2.

El Agente Interfaz proporciona al usuario, un conjunto de interfaces que incluyen herramientas para informarlo sobre su estado físico, requerimientos nutricionales y gestionar sus automonitoreos de glucosa, presión arterial y actividad física, así como una lista de alimentos para ser considerados en la recomendación alimenticia.

El Agente Dieta es el encargado de crear la recomendación alimenticia, esta es generada mediante un algortimo de inteligencia artifical, en particular, un algoritmo genético. Este se encarga de crear un conjunto de individuos (soluciones posibles al problema) expresados como cromosomas que posteriormente pasarán por acciones semejantes a la evolución biológica (mutaciones y recombinaciones genéticas), así también a una selección de acuerdo a una función *fitness*, que decide cuáles son los individuos más adaptados, que sobreviven, y cuáles son los menos aptos, que son descartados, hasta encontrar la solución que más se adapte a los requerimientos nutricionales. A este modelo de algoritmo genético se le aplicó una prueba t de student dando como resultado un 94-98% de efectividad respecto al requerimiento nutricional y

una prueba de calidad exponencial con un resultado de 99% de recomendaciones alimenticias acertadas esto quiere decir que cumplen con los requisitos nutricionales, durante un tiempo de 30 días.

Por último, el Agente Evaluación se encarga de juzgar la efectividad de la recomendación alimenticia proporcionada mediante un control difuso, tomando en cuenta los factores de regulación de glucosa y presión arterial generados por los automonitoreos preprandial y postprandial (antes y después de comer), dependiendo del valor de este factor de regulación, el control da un veredicto de qué tan efectiva fue la recomendación propuesta.

ÍNDICE

Capítulo I 1
1.1 Introducción. 2
1.2 Descripción del problema. 3
1.3 Propuesta de solución. 6
1.4 Objetivo general. 7
1.5 Objetivos específicos. 7
Capítulo II 8
2.1 Estado del arte. 9
2.2 Solución inteligente. 9
2.3 Dependiente del proveedor de servicios de salud. 12
Capítulo III 20
3.1 Descripción del trabajo. 21
3.2 Agente Interfaz. 21
3.2.1 Expediente Personal. 22
3.2.2 Evaluación antropométrica. 23
3.2.3 Evaluación bioquímica. 24
3.2.4 Requerimientos Nutricionales. 24
3.2.5 Preferencias Alimenticias. 27
3.3 Agente dieta 28
3.3.1 Algoritmo Genético. 29
3.4 Agente Evaluación. 33
3.4.1 Variables de estado. 35
3.4.2 Variables de control. 36
3.4.3 Reglas Difusas. 37
3.4.4 Parámetros de la inferencia difusa. 37
3.5 Comunicación entre agentes 38
Capítulo IV 40
4.1 Procedimiento y descripción de actividades 42
4.1.1 Agente Interfaz 42
4.1.1.1 Acceso 42
4.1.1.2 Panel de control Usuario 43
4.1.1.2.1 Usuario 43
4.1.1.1.2 Monitoreo 43
4.1.1.3 Panel de Control Plan Dieta 45
4.1.2 Agente Dieta 47
4.1.3 Agente Evaluación 48
4.2 Análisis y resultados 49
4.2.1 Análisis de la variable dependiente. 49
4.2.2 Análisis de la variable independiente. 51
4.2.3 Prueba del Control difuso. 52
4.2.3.1 Fuzzyficación (Singleton) 53
4.2.3.2 Reglas Activadas 53
4.2.3.4 Modus ponens Difuso 53
4.3.2.5 Agregación y defuzzyficación 54
5.1 Conclusiones 56

5.2 Trabajo a Futuro 58
5.3 Referencias bibliográficas 60
Anexos 63
A.1 Programas. 63
A.1.1 Creación y comunicación de los Agentes. 63
A.1.2 Agente Interfaz 64
A.1.2.1 Acceso. 64
A.1.2.2 Panel de Control Usuario 69
A.1.2.3 Panel de Control Dieta 84
A.1.2.4 Evaluación Antropométrica, Bioquímica y Nutricional. 97
A.1.2.5 Alimento 99
A.1.3 Agente Dieta 102
A.1.3.1 Clase Principal del Algoritmo genético. 102
A.1.3.2 Función fitness. 106
A.1.4 Agente Evaluación 110
A.1.4.1 Control difuso. 110

ÍNDICE DE TABLAS

Tabla 1. Tabla comparativa de trabajos relacionados. 12
Tabla 2. Índices de Glucemia. 24
Tabla 3. Índices de Presión Arterial. 24
Tabla 4. Índices Aceptables de Actividad Física. 24
Tabla 5. Cromosoma de ejemplo. 30
Tabla 6. Pruebas del algoritmo genético. 50
Tabla 7. Escala de efectividad del modelo de algoritmo genético. 51

ÍNDICE DE FIGURAS

Figura 1. Modelo del sistema multiagente. 21
Figura 2. Base de datos del expediente personal. 22
Figura 3. Ejemplo de crossover del algoritmo genético. 33
Figura 4. Funciones de membresía del factor de regulación de glucosa. 35
Figura 5. Funciones de membresía del factor de regulación de presión arterial. 36
Figura 6. Funciones de membresía de la evaluación. 37
Figura 7. Fuzzy Asocciation Matrix de la evaluación. 37
Figura 8. Diagrama de comunicación del sistema multiagente. 38
Figura 9. Ventana de Acceso y Crear usuario. 42
Figura 10. Panel de control usuario. 43
Figura 11. Pestaña monitoreo del Panel de control usuario. 44
Figura 12. Alertas de Glucosa y Presión arterial. 44
Figura 13. Botón Agente Evalucación 45
Figura 14. Panel de Control dieta. 46
Figura 15. Sección Comidas ADD. 46
Figura 16. Botón Agente Dieta. 47
Figura 17. Representación de la recomendación alimenticia. 48
Figura 18. Regiones de aceptación y rechazo en el contraste de hipótesis. 50
Figura 19. Escenario de prueba. 52

Figura 20. Valor singleton del Factor de regulación de glucosa.....53
Figura 21. Valor singleton del Factor de regulación de presión arterial. 53
Figura 22. Reglas Activas.....53
Figura 23. Valor de la variable de control evaluación.....54
Figura 24. Curvas de conmutación del sistema de inferencia difuso. 54

Capítulo I

1.1 INTRODUCCIÓN.

Durante las últimas décadas México a generado y recibido los beneficios de una mejoría notable en las condiciones de salud. Las principales causas de muerte se han modificado. Las infecciones comunes y los problemas que tienen vínculos con la desnutrición y la reproducción han sido desplazados por las enfermedades no transmisibles y las lesiones, que hoy concentran más del 85% de las causas de muerte en el país [1].

En este contexto, las enfermedades no transmisibles requieren ahora de tratamientos más complejos y prolongados que los de otra índole; por lo tanto su manejo es más costoso y significan una mayor carga económica para la sociedad. Las enfermedades crónicas no trasmisibles (ECNT) son enfermedades de larga duración y por lo general de progresión lenta, entre ellas se encuentran: las enfermedades cardíacas, el cáncer, las enfermedades respiratorias y la diabetes.

En particular, la diabetes es un problema mundial. En México la incidencia, prevalencia y mortalidad están incrementándose a un ritmo acelerado. La diabetes se está mostrando en etapas de la vida cada vez más tempranas, con el consecuente incremento de las complicaciones, además de su mayor frecuencia también se presentan en población más joven [2].

La diabetes mellitus o diabetes es una enfermedad crónica degenerativa que se presenta cuando el páncreas no produce insulina, o bien, la que produce no es utilizada de manera eficiente por el organismo; la insulina es la responsable de que la glucosa de los alimentos sea absorbida por las células y proporcionar de energía al organismo [3].

Existen dos principales tipos de diabetes [3]:

- Diabetes tipo I. Dependiente de la insulina, a veces se le llama diabetes juvenil, porque normalmente comienza durante la infancia. Como el cuerpo no produce insulina la persona debe inyectarse insulina para vivir. Menos del 10% de los afectados padecen de diabetes tipo I.
- Diabetes tipo II. Surge en adultos, el cuerpo si crea insulina, o bien no produce lo suficiente, o no puede aprovechar la que fabrica.

Para los profesionales de la salud y la industria farmacéutica, se trata de un momento difícil, debido a que los sistemas de salud se han convertido en una sobrecarga, y la capacidad de operar efectivamente dentro de los sistemas actuales se está convirtiendo en inadecuada. Las soluciones de salud móvil (mHealth) ayudan a hacer frente a estos desafíos. Por ejemplo, para los profesionales de la Salud, ¿cómo hacer frente a las crecientes demandas de su tiempo y el creciente número de pacientes con enfermedad crónica que implica el seguimiento a largo plazo y la atención?, ¿Cómo se mantiene al día con los avances médicos para que sus pacientes reciban la mejor atención posible sea cual sea su ubicación, cualesquiera que sean sus recursos, sean cuales sean sus necesidades?. En términos generales, la salud móvil se refiere a la utilización de la comunicación móvil y dispositivos para la prestación de servicios de salud para los pacientes, profesionales sanitarios y cuidadores. Investigaciones han demostrado claramente que las soluciones mHealth podrían ofrecer una excelente oportunidad para los profesionales sanitarios y la industria farmacéutica para medrar dramáticamente el cuidado de la salud en todo el mundo, mejorando la calidad de atención reduciendo la carga de trabajo, ayudando en la gestión de las enfermedades crónicas y haciendo asequibles los costos de ascistencia sanitaria [4].

La diabetes mellitus tipo II representa altos costos para el individuo y la sociedad, la mayoría de estos costos se derivan de varias complicaciones que se pueden reducir, aplazar e incluso prevenir si se controla la enfermedad. La declaración de la Organización Mundial de la Salud y la Organización Panamericana de la Salud pone de relieve la importancia cada vez mayor que tiene la diabetes como causa de morbilidad y mortalidad de la población. Las comunidades deben promover la alimentación saludable y el ejercicio físico con objeto de prevenir la enfermedad [36].

1.2 DESCRIPCIÓN DEL PROBLEMA.

La diabetes es una enfermedad que dura toda la vida (crónica) en la cual hay altos niveles de azúcar (glucosa) en la sangre. La diabetes tipo 2 es la forma más común de esta enfermedad. Por lo general la diabetes tipo 2 se desarrolla lentamente con el tiempo, podrían pasar años hasta que los síntomas aparezcan o se reconoscan, tiempo durante el cual el organismo se va deteriorando debido al exceso de glucosa en la sangre. La mayoría de las personas con esta enfermedad tienen sobrepeso en el momento del

diagnóstico. El aumento de la grasa le dificulta al cuerpo el uso de la insulina de manera correcta [37].

En México el comportamiento de la diabetes mellitus tipo 2, de 1998 al 2012 se ha observado una tendencia hacia el incremento en un 4.7%, pasando de una tasa de mortalidad de 342.1 a 358.2 casos por cada 100 mil habitantes, específicamente en el año 2012 se reportaron 418,797 pacientes diagnosticados con diabetes, si la tendencia permanece igual se espera para el año 2030 un aumento del 37.8% en el número de casos y 23.9% en la tasa de morbilidad [5].

Los antecedentes familiares y los genes juegan un papel importante en la diabetes tipo 2. Un bajo nivel de actividad física, una dieta deficiente (que no cumple los requerimientos nutricionales necesarios) y el sobrepeso, aumentan el riesgo de padecer diabetes. Un paciente con diabetes tipo II debe ser responsable de llevar una alimentación balanceada respecto a su estado de salud, sumado a un régimen de ejercicio que le permita mantener su peso estable para evitar complicaciones derivadas con el sobrepeso. Otro aspecto importante en el cuidado de la diabetes es la hipertensión, estas dos, el sobrepeso y la hipertensión, son las principales factores de riesgo cardiovascular [37].

Las aplicaciones mHealth, cuando una persona padece diabetes tipo 2, intentan monitorear de manera constante los niveles de glucosa en sangre, presión arterial y peso, mediante dispositivos de medición portátil, por qué de no ser así, puede presentarse una elevación extrema en cualquiera de los tres factores antes mencionados sin darse cuenta, y esto puede resultar peligroso.

La glucosa es el principal problema en pacientes con diabetes, hacer una medición de esta, es un elemento fundamental para cualquier plan de tratamiento de la diabetes. El automonitoreo periódico ofrece información sobre la eficacia del plan de tratamiento diario para controlar los niveles de glucemia [3]. Con esto el paciente y el médico pueden evaluar los resultados para determinar la necesidad de ajustes en el tratamiento y ayudar así a lograr un control eficiente de la diabetes.

Por otra parte, hacer suficiente actividad física de manera regular (30-40 min 5 días a la semana) es importante para mantener una buena salud y asegurar un buen

control de la diabetes. Esto resulta en beneficios como mejorar la respuesta del cuerpo a la insulina, bajar la presión sanguínea, controlar el peso y reducir el riesgo de sufrir complicaciones de la diabetes. A pesar de la importancia conocida del ejercicio físico en relación con la salud mental, la mayoría de las personas con diabetes no cumplen la cantidad necesaria de actividad física. Como resultado de ello sigue siendo un desafío promover la ejercicio físico entre los pacientes con DM2 [38].

En pacientes con diabetes la presión alta afecta a un 40-60% de la población con diabetes esto contribuye al desarrollo y la progresión de las complicaciones propias de esta enfermedad metabólica. En la diabetes tipo 2, la hipertensión puede estar presente en el momento del diagnóstico o incluso antes de desarrollarse, y a menudo es parte de un síndrome que incluye intolerancia a la glucosa, resistencia a la insulina y obesidad. Las personas con diabetes no pueden tratar la hipertensión con cualquier medicamento, ya que puede afectar a su tratamiento diabetológico [3].

Determinar las necesidades nutricionales en un paciente con DM2, proporcionando una recomendación de acuerdo a sus necesidades específicas, y finalmente la combinación de los principios por ejemplo las de la dieta American Diabetes Association (ADA) y las preferencias nutricionales personales, puede ser una tarea difícil, pues las preferencias personales, en ocasiones, no proporcionan los nutrientes necesarios para el control de la DM2. En la investigación StudiesExplained [6], se llega a la conclusión de que las necesidades de nutrición de los pacientes a menudo difieren de las proporcionadas en el hogar, en comparación con la configuración proporcionada en hospitales, debido a que los nutricionistas desempeñan un papel significativo en la gestión de la atención de salud en los pacientes con DM2, pues proporcionan los conocimientos y habilidades en la terapia médica nutricional[39].

La DM2 es progresiva y crónica por lo tanto el paciente es dependiente de su proveedor de servicios tales como la familia y los servicios locales de salud [6].Los pacientes deben controlar grandes cantidades de información provenientes de su automonitoreo de glucosa y presión arterial, estos datos son analizados por parte de los médicos para tomar una decisión respecto a su alimentación y ejercicio físico.

Debido a esto, uno de los retos es proporcionar el apoyo necesario a los pacientes con diabetes, en etapas iniciales y a largo plazo que permitan su autocuidado en formas factibles y rentables. Las aplicaciones de apoyo a la toma de decisiones en el dominio de cuidado de la salud son usadas para ayudar al personal sanitario a completar su trabajo con mayor precisión y también mejorar la calidad de la atención al paciente.

Es por esto que en el presente trabajo se propone el desarrollo de un sistema multiagente, que apoye a la toma de decisiones mediante la recomendación de un plan alimenticio para un paciente con DM2 considerando sus factores de glucosa, presión arterial, actividad física y preferencias alimenticias.

1.3 PROPUESTA DE SOLUCIÓN.

En este trabajo se plantea una solución basada en un sistema multiagente, el cuál proporciona un plan alimenticio personalizado para pacientes con DM2. El sistema está compuesto específicamente por tres agentes: Agente Interfaz, Agente Dieta y Agente Evaluación.

El Agente Interfaz es el encargado de la recolección y almacenamiento de los datos antropométricos, bioquímicos y preferencias alimenticias proporcionados por el paciente para realizar una evaluación antropométrica y química, estas dos evaluaciones proporcionan el estado físico del paciente y el requerimiento nutricional para su recomendación alimenticia.

El Agente Dieta realiza la tarea de calcular y proporcionar una recomendación alimenticia respecto a los datos de análisis proporcionados por el Agente Interfaz, mediante un algoritmo genético que se encarga de combinar estos alimentos de tal manera que cumplan con los requerimientos nutricionales del paciente. Y por último el Agente Evaluación por medio de un control difuso, es el encargado de evaluar la recomendación dada respecto a los factores de regulación de glucosa y presión arterial del paciente durante la utilización de la recomendación alimenticia.

El trabajo conjunto de estos tres agentes pretende solucionar los problemas de déficit alimentico presentes en la mayoría de los pacientes con DM2 con el propósito de regular sus niveles de glucosa y presión arterial.

1.4 OBJETIVO GENERAL.

Desarrollar un sistema multiagente para el monitoreo de pacientes con diabetes mellitus tipo II, con el propósito de elaborar un plan alimenticio y la generación de alertas para evitar complicaciones relacionadas con este padecimiento.

1.5 OBJETIVOS ESPECÍFICOS.

- Implementar un agente que contenga una interfaz gráfica, para almacenar los datos del automonitoreo de la glucosa y la presión arterial, así como las preferencias alimenticias a considerar en la recomendación.
- Desarrollar una base de datos para el almacenamiento del automonitoreo con el fin de crear un historial clínico del paciente.
- Implementar un agente que procese los datos almacenados proporcionados del paciente para la elaboración de una recomendación alimenticia mediante un algoritmo genético.
- Implementar un agente, que evalúe la recomendación alimenticia, por medio de un control difuso.
- Emitir alertas para el automonitoreo, en el caso que alguna de las lecturas alcance un rango peligroso, para evitar complicaciones relacionadas con la diabetes tipo II.

Capítulo II

2.1 ESTADO DEL ARTE.

La salud móvil (mHealth) tiene el potencial de cambiar la forma en que la asistencia sanitaria es entregada por los profesionales de la salud a los pacientes con enfermedades crónicas no transmisibles para aumentar la eficiencia, la rentabilidad de la atención, proporcionar a los pacientes un control más informado de su condición, cambiar los comportamientos que perpetúan la enfermedad y ayudar en la prevención de enfermedades crónicas.

En particular, el tratamiento de la diabetes mellitus tipo 2 mediante dispositivos móviles considera tres aspectos principales de este padecimiento, la glucosa, la presión arterial y la actividad física. A partir de esto las técnicas se clasificaron en dos, las que platean una solución inteligente y las que siguen siendo dependientes de su proveedor de servicios de salud.

2.2 SOLUCIÓN INTELIGENTE.

Este tipo de trabajos se encargan de administrar el volumen de información proveniente del automonitoreo del paciente, para su posterior almacenamiento y análisis mediante algoritmos de inteligencia artificial, con el propósito de modificar la dosis de insulina, o el plan alimenticio del paciente. La ventaja de estos trabajos es que un paciente con diabetes tipo 2 deja de ser dependiente de su proveedor de servicios de salud, el automonitoreo puede ser realizado en cualquier lugar mediante sensores portátiles y obtiene la capacidad de llevar el control de su enfermedad mediante un dispositivo móvil. Las desventajas de este tipo de trabajos es que dan solución a un aspecto en particular de esta enfermedad, una recomendación propuesta para un problema específico ya sea la hipertensión, la glucosa o la actividad física. Existen pocas soluciones inteligentes que integren estos tres aspectos para dar una recomendación general a este padecimiento.

Lee (2008) et al. En [7] desarrollaron un agente inteligente basado en una ontología para la recomendación alimenticia para pacientes con DM2. El agente crea un plan alimenticio dependiendo de sus gustos, estilo de vida y necesidades de salud que es una ontología personal. El agente para proporcionar una recomendación utiliza dos ontologías una de comida taiwanesa y otra de comida personal. El mecanismo de

inferencia difusa recupera los registros de la ontología personal tomando como entradas principales los carbohidratos, proteínas y grasas para recomendar un plan alimenticio. La ontología fue probada en 8 voluntarios con diabetes, los registros de comidas fueron recogidos por la plataforma de gestión de salud durante 5 meses. Los resultados experimentales muestran que el agente propuesto es capaz de recomenda un menú de cena individual de acuerdo con a ontología de dominio preconstruida.

Kafali (2013) et al. En [8] implementaron un sistema multiagente, está formado por un ambiente de dispositivos portátiles para controlar y comunicar parámetros fisiológicos y otro contexto relacionado con la salud del paciente como la actividad física y señales vitales del cuerpo. Los datos son interpretados por agentes inteligentes que utilizan un conocimiento experto para derivar información sobre el estado diabético de la persona, que luego es interpretado mediante conocimiento biomédico experto para posteriormente ser representado en el dispositivo o por profesionales que mediante servicios web ayudan al tratamiento, diagnóstico y la vida del hombre en cuestión. El sistema multiagente en este caso no ha sido evaluado, pero se especifica en el trabajo a futuro analizarlo mediante pruebas de usabilidad.

Campos-Delgado (2006) et al. En [9] desarrollaron un algoritmo de asesoramiento/control para la diabetes tipo 1 bajo un tratamiento insulínico intensivo basado en un régimen de inyecciones diarias múltiples. El algoritmo incorpora un conocimiento experto sobre el tratamiento de esta enfermedad mediante el uso de un control difuso Mam-dani para regular el nivel de glucosa en la sangre. La estrategia de control se basa en 2 iteraciones para superar la variabilidad dinámica de la glucosa. La primera iteración ofrece la cantidad de insulina para formulaciones rápidas/cortas e intermedias/insulina de acción prolongada programadas en 3 aplicaciones antes de las comidas. La segunda iteración ajusta los importes máximos de insulina proporcionados al paciente en una escala de días, este trabaja como controlador de la primera iteración. El sistema fue probado mediante una simulación compuesta por 4 escenarios: Cuando el paciente produce pequeñas y altas docis de insulina, variando la ingesta de carbohidratos, variación drástica de insulina y glucosa y la ultima semejante a la primera con la variación de que el sensor de glucosa proporciona mediciones cada 30 min, por lo tanto se espera una recomendación más precisa. Los resultados de la simulación demuestran

la eficacia y robustez de el algortimo en la regulación de glucosa en la sangre, además puede ser implementado en un microordenador para un tratamiento casero.

Quinn CC. (2008) et al. En [10] desarrollaron un software basado en un dispositivo móvil diseñado por endocrinólogos. El software proporciona información en tiempo real sobre los pacientes, nivel de glucosa, regímenes de medicación, los algoritmos de hiper e hipo glucemia incorporados y solicita datos adicionales necesarios para evaluar la gestión de la diabetes. Los datos del paciente son capturados y transferidos a servidores seguros donde son analizados por algoritmos estadísticos propietarios. El sistema envía libros de registros generados con planes de tratamiento sugeridos a los proveedores de salud encargados de los pacientes. El software fue probado en 15 pacientes donde se demostró que las decisiones del sistema facilitan el tratamiento, proporcionando datos organizados y reduciendo el tiempo de revisión del cuaderno de bitácora del paciente.

Schumann (2012) et al. En [11] desarrollaron una aplicación basada en cliente-servidor, donde un sistema multiagente analiza los datos de los pacientes y controla su condición. Los agentes se tratan como recursos persistentes asociados a un paciente, cada que el paciente inicia sesión en el sistema el estado del agente se reanuda desde una base de datos de agente. Esto les permite manejar un gran número de pacientes. La aplicación fue porbada en tres escenarios: considerando que el paciente tiene un mal control glucémico, cuando el paciente presenta síntomas de preeclampsia (hipertensión arterial y proteína en la orina en mujeres embarazadas) y un paciente en etapa final de embarazo, con un pobre control glucémico y demasiado aumento de peso. Bajo las condiciones anteiores se demostró que el sistema tiene una base sólida y mejora la atención a las mujeres embarazadas y a sus bebes.

Kurran (2010) et al. En [12] desarrollaron una solución en forma de una red neuronal inteligente que se ejecuta en dispositivos móviles. Los pacientes pueden ejecutar la aplicación en tiempo real a través de una interfaz gráfica de usuario. La red neuronal consta de 4 neuronas. La primera es la glucosa. Si el nivel de glucosa es alto, entonces habrá un valor positivo multiplicado por el peso, lo que implica una cantidad positiva de insulina para ser inyectada. Si el nivel de la glucosa del usuario es baja, entonces los pesos se multiplican por un valor negativo, lo que resulta en una disminución de la dosis total de insulina. Un ensayo se llevo a cabo para evaluar la efectividad de la

solución durante dos semanas con 6 pacientes. Los principales objetivos fueron inivestigar la interfaz de usuario, porbar el mando a distancia, el envío de datos a través de una red 3G y el registro de los datos para su posterior tratamiento por a red neuronal. Dando como resultado que su enfoque (que tan bien puede ser conocido como insulinoterapia intensiva) tiene implicaciones en futuras bombas inteligentes de insulina.

El presente trabajo se encuentra posicionado en este grupo como lo muestra la tabla 1, proporcionando una recomendación alimenticia al paciente mediante un sistema multiagente y algoritmos de inteligencia artificial en este caso un algoritmo genético y un control difuso para solucionar los problemas nutricionales del paciente y con ello regular su glucosa y presión arterial.

	Autor					
Factor a considerar	Ö. Kafali 2013 [8]	C. Lee 2008 [7]	R. Schuman 2012 [11]	K. Kurran 2010 [12]	S. Bin-Sabbhar 2012 [13]	O. Lagunes 2014 [35]
Glucosa	X	X	X	X	X	X
Presión Arterial	X	-	-	-	-	X
Actividad Física	X	-	-	-	-	X
Recomendación	Farmacológico	Nutricional	Farmacológico	Farmacológico	Farmacológico	Nutricional

Tabla 1. Tabla comparativa de trabajos relacionados.

2.3 DEPENDIENTE DEL PROVEEDOR DE SERVICIOS DE SALUD.

Plantean una solución mediante sistema de mensajes cortos (SMS), o servicios web, donde el paciente ingresa los datos de su automonitoreo a una base de datos central donde el médico responsable del tratamiento puede acceder a los datos ingresados, analizarlos y de esta manera modificar el tratamiento del paciente, lo que representa una ventaja. La desventaja de una solución como esta, es que el paciente por incertidumbre recurre a la automedicación o simplemente decide no hacerse responsable de su enfermedad, ya que las visitas a su médico cuando una persona padece de diabetes deben ser constantes. Esto da la impresión de una pérdida de tiempo y dinero. Por otra parte, al paciente le resulta laborioso llevar un control personal de su padecimiento debido a la frecuencia de sus automonitoreos pues es necesario un conocimiento de las causas y consecuencias.

Bin-Shabbar (2012) et al. En [13] implementaron un sistema móvil formado por 3 partes: paciente (expediente), endocrinólogo (care center), examen médico mensual (montly services). Este sistema provee un componente de monitoreo diario de glucosa al paciente, este monitoreo se almacena en la base de datos central, a partir de esto, el endocrinólogo puede acceder a la aplicación y solicitar el informe de un paciente determinado, puede modificar la dosis de insulina y el plan de tratamiento si es necesario. Evaluaron la efectividad de la aplicación, dependiendo la facilidad con la que usuario es capaz de alcanzar el objetivo de obtener un plan de tratamiento insulínico y su usabilidad considerando que el usuario puede tene un problema de vista o algún otro problema físico que dificulte el manejo del sistema.

Årsand (2010) et al. En [14] propusieron una aplicación móvil que facilita el almacenamiento de los automonitoreos de un paciente con DM2, en este caso: glucosa, actividad física y hábitos alimenticios. El análisis de lo datos se realiza seleccionando el alimento que se ingirió antes de cada medición de la glucosa. Estos hábitos se modifican mostrando al usuario los grupos alimenticios clasificados como bocadillos, comidas y bebidas altas y bajas en carbohidratos. La aplicación se probó con varios usuarios durante 6 meses, demostrando una buena usabilidad y ajustando en varios pacientes la medicación, hábitos alimenticios y actividad física.

Van der Weegen (2012) et al. En [15] desarrollaron un sistema de monitoreo móvil, plantea una solución para el sedentarismo en pacientes con DM2. Esto se realiza por medio de 3 feedbacks. El primero mediante el automonitoreo de la actividad física mediante un acelerómetro 3D. La información proveniente de este se muestra en un gráfico de barras indicando cuál es el objetivo a alcanzar. El segundo se envía la información a un sitio web donde los usuarios del sistema comentan sobre la actividad física realizada y la meta lograda. Por último el tercer feedback los datos se envían al proveedor de salud responsable de la actividad física para su análisis y modificación. El estúdio realizado a esta aplicación se realizó recopilando las condiciones de un usuario con DM2, para evitar presuposiciones. La solución demostró hacer conciencia en el usuario de su inactividad motivándolo a cambiar su comportamiento respecto a su actividad física.

López (2009) et al. En [16] implementaron un sistema móvil, para la captura del automonitoreo de glucosa y presión arterial. Estos datos son enviados a una base de datos central donde un endocrinólogo es capaz de acceder a estos datos y desde ahí modificar el tratamiento, enviar alertas. La evaluación de esta solución se realizo mediante una encuesta en médicos entre 35-64 años y paciente 55-84 años demostrando altos problemas de usabilidad. Las percepciones del médico y el paciente respecto al sistema no son del todo positivos respecto a la facilidad de uso y la satisfacción.

Pandey (2012) et al. En [17] desarrollaron un sistema móvil, para el automanejo de la DM2. Permite la comunicación constante con el proveedor de servicios de salud mediante SMS, el paciente puede enviar distintos datos glucosa, presión arterial, dosis de insulina entre otros datos, también el paciente puede solicitar y mostrar mediciones anteriores, cambiar la hora y la frecuencia de las alertas así como acceder a material educativo multimedia sobre la DM2 y su tratamiento. El sistema fue probado en la clínica para DM2 de Lucknow. 5 pacientes con DM2 usaron la aplicación durante 2 semanas, junto a su médico. Durante esta etapa 220 mensajes entrantes fueron enviados al médico y 20 mensajes salientes fueron enviados de vuelta a los pacientes. Después de las pruebas, los niveles de HbAc1 se compararon antes y después del estudio. A pesar de que el resultado no demuestra una mejora significativa en la condición médica de los pacientes, reportaron una satisfacción con el uso del sistema al sentirse conectados con su médico.

H Yu et (2012) al. En [18] desarrollaron un sistema web para la gestión de la DM2. El estudio consta de 5 fases: desarrollo de la intervención, pruebas de viabilidad, las pruebas de usabilidad, el perfeccionamiento de la intervención y la evaluación de la intervención usando métodos mixtos. El sistema posee varios componentes que ayudan al paciente a llevar el control de su padecimiento mediante la retroalimentación de información ya sea paciente a paciente o paciente a médico. Notificación por correo electrónico, comics educativos, videos educativos, diario web, etc. El estudió realizado a la aplicación se divide en 4 fases: Desarrollo de la intervención que verifica la eficacia de las herramientas web proporcionadas al paciente, pruebas de viabilidad usando una metodología de grupos focales con personas mayores a 25 años de diferentes etnias, pruebas de usabilidad utilizando un análisis cognitivo de tareas que representan escenarios para cubrir las principales funcionaleidades de la aplicación y refinamiento de

la intervención basado en los datos de la prueba de viabilidad se realizaron mejoras al sitio web. Basado en las evidencias de las 4 estrategias de análisis mencionadas anteriormente, se llegó a la conlcusión, que el sitio web está diseñado óptimamente para mejorar los resultados psicológicos, clínicos y complemetar la prestación de atención de la salud.

Samaleh (2012) et al. En [19] desarrollaron un sistema cliente-servidor. Del lado del servidor consiste en dos módulos, un modulo web y un Gateway SMS. El cliente es una aplicación dedicada, esta también consiste en dos módulos: un módulo core y uno E-Learning. Ambos el paciente y el médico pueden comunicarse. El paciente debe enviar regularmente SMS con la información de su automonitoreo. Estos mensajes son recibidos por un modem GSM entonces el médico puede iniciar comunicación mediante la interfaz web enviando SMS al paciente. La solución fue probada en una clínica, durante esta etapa 150 mensajes fueron enviados al médico y 10 mensajes fueron enviados de vuelta a los pacientes. A pesar de que el resultado no demuestra una mejora significativa en la condición médica de los pacientes, reportaron una satisfacción con el uso del sistema al sentirse conectados con su médico.

Blanchet (2008) et al. En [20] propusieron un proyecto de telemedicina en lo que respecta a la atención domiciliaria de la diabetes. La parte central o Home Telemedicine Unit (HTU) del proyecto desarrollada por American Telecare consta de 4 funciones: teleconferencia síncrona a través de líneas telefónicas estándar, transmisión electrónica segura del automonitoreo de la glucosa, revisión segura de datos clínicos mediante mensajes web y el acceso a materiales educativos basados en web. El sistema fue diseñado para que personas mayores y personas con poca o nula experiencia con el manejo de computadoras, puedan manejarlo de manera eficaz. El proyecto se puso en marcha en dicembre del año 2000, con una cantidad de 1500 pacientes, el proyecto a generado numerosos. Todos los pacientes han mostrado mejoras a lo largo del estudio, en lo que respecta a sus mediciones de HbAc1, colesterol y la presión arterial.

Sahar (2013) et al. En [21] desarrollaron un sistema de apoyo en la toma de decisiones basado en web, para pacientes con DM2. El sistema proporciona una decisión respecto a la alimentación y rutinas de ejercicios. Utilizando como variables principales la estatura y el peso para calcular el índice de masa corporal. Después de esto ya sea en la

sección de “planeador de comidas” o “actividades”, el usuario puede experimentar diferentes alternativas de ejercicio y comidas. Hecho esto el sistema mostrará las calorías y la predicción resultante. El análisis hecho a la aplicación, demuestra que puede facilitar el tratamiento para pacientes con DM2, dándoles autonomía y autogestión a largo plazo. También presenta algunas limitaciones, por ejemplo, es difícil cubrir todos los alimentos dietéticos y también su importe exacto de nutrientes.

HealthPia, Inc. (2007) En [22] desarrollaron un prototipo de sistema integrando un monitor de glucosa en un teléfono celular convencional. El sistema recolecta los automonitoreos de glucosa del paciente y los muestra en pantalla; después de esto los datos son transmitidos a un servidor donde a través de un sitio web, los pacientes, parientes y médicos pueden ver los valores de la glucosa bajo una conexión de internet segura. El usuario también puede utilizar el teléfono celular para discutir las opciones terapéuticas con su proveedor de servicios de salud. El estudio realizado a esta solución se realizo en pacientes adolescentes y se demostró, que a los pacientes les gusta la integración de estas dos tecnologías y acordaron que el glucómetro es fácil de usar y que la herramienta era útil para el manejo de su diabetes.

Ferrer-Roca (2004) et al. En [23] implementaron un sistema basado en web, capaz de recibir y mostrar datos del paciente con diabetes. Utilizando una interfaz web el médico puede revisar estos datos y enviar mensajes a los pacientes. Los datos se almacenan en un servidor usando una base de datos. Cada mes el sistema calcula un valor medio de glucosa en sangre para cada paciente. El estudio experimental al sistema, consistió en 77 pacientes con DM2 durante 3 meses, el hallazgo de este estudio mostró que la intervención mediante SMS tuvieron cambios significativos en su HbAc1 con un cambio medio de -1.01%.

Franklin (2003) et al. En [24] desarrollaron una red de apoyo, basado en un sistema de mensajes de texto diseñado para entregar los mensajes dirigidos individualmente a la información general de la diabetes. Estrategias basadas en la motivación, teoría basada en la cognición social, el modelo de creencias de salud y el establecimiento de metas forman la base teórica del contenido del mensaje. El sistema ofrece un contacto intermedio y apoyo entre visitas a la clínica. En el momento de la evaluación de esta solución, demostró una tasa de aceptación del 73% de los participantes.

Hanahuer (2004) et al. En [25] desarrollaron un sistema de recordatorios automáticos mediante email o SMS para ayudar al control optimo glucémico en pacientes con diabetes. El usuario crea en el sistema un calendario de acuerdo a su preferencia. El usuario responde al recordatorio mediante el automonitoreo de glucosa. La medición puede ser consultada y modificada desde un sitio web. Los resultados preliminares de 10 usuarios de edades entre 18 y 20 años en un periodo de 3 meses, indican que presentaron un promedio de 1.3 y 2.7% de regulación en la glucosa, debido a esto no se obtuvieron resultados significativos.

Rami (2006) et al. En [26] desarrollaron un sistema de telemedicina de apoyo sobre el control glucémico en adolecentes con diabetes mellitus tipo 1. Los pacientes envía sus datos (fecha, hora, glucosa, la ingesta de carbohidratos, la dosis de insulina) a través de un teléfono móvil, al menos diariamente, a un servidor donde los endocrinólogos devuelven sus recomendaciones mediante SMS una vez a la semana. La evaluación realizada a la aplicación mostró una mejora en el control glucémico de los usuarios. Los pacientes clasificaron el programa como una buena idea para la gestión de la DM2. Problemas técnicos con el envio de los SMS llevaron a la perdida de datos y la disminución de la satisfacción del pacientel.

Rossi (2010) et al. En [27] desarrollaron una herramienta que incorpora distintas funciones; una calculadora de carbohidratos/insulina, un dispositivo tecnológico de información y un sistema de telemedicina basado en la comunicación de un profesional de la salud con un paciente a través de SMS. Se permite a los pacientes manejar una dieta flexible y calcular la insulina contenida en cada comida. Además se incluye un algoritmo para el cálculo de dosis de insulina basado en valores de glucemia en ayunas y la presencia de episodios de hipoglucemia. El estudio de esta herramienta se realizó en 130 pacientes con edades entre 9 y 35 años en un tiempo de 6 hrs y demostró diferencias significativas en la lectura de glucemia en ayunas y peso corporal.

Bell (2012) et al. En [28] desarrollaron un sistema para personas con diabetes, mediante el envió de videos diariamente con consejos relacionados con la diabetes y recordatorios entregados a través de teléfonos celulares. La intervención fue un complemento a la atención habitual, que busca proporcionar apoyo generalizado de estilo de vida a personas que no estaban cumpliendo con los objetivos glucémicos a pesar de

recibir cuidado especializado de la diabetes. El paciente recibió durante 6 meses diariamente a su teléfono móvil videos sobre el autocuidado de la diabetes para mejorar su control glucémico. Los pacientes demostraron una disminución del 0.6% en la HbAc1 lo cual indíca un control aceptable de la DM2.

Mckay (2002) et al. En [29] desarrollaron una red de diabetes (D-Net), es una prueba aleatoria de automanejo de la diabetes y soporte de intervención entre pares mediante un sitio web. La intervención de autogestión se llevó a cabo a través de internet mediante la educación en el cuidado de la diabetes con tres objetivos principales: aumento de los conocimientos, proporcionar capacitación y mejorar el apoyo social. Esto se logra proporcionando al paciente artículos sobre estilo de vida, aspectos médicos y nutricionales de la diabetes. También el paciente cuenta con asesoramiento diabético de un profesional a través de internet el cuál evalúa su desempeño durante la utilización del sistema. Los participantes, para el evaluación de este sistema, fueron 75 hombres y 85 mujeres que fueron diagnosticados con DM2, los participantes tenían una edad media de 59 años, y realizaron la prueba durante durante 3 meses, no se demostraron efectos significativos en la reducción del colesterol ni la HbAc1.

Gammon (2005) et al. En [30] diseñó un prototipo para transferir automáticamente las lecturas del monitor de glucosa en la sangre de un niño al teléfono móvil de su padre. Esto se realiza programando un teléfono móvil para que envié automáticamente un mensaje de texto (SMS), siempre y cuando el teléfono este al alcance de 10m del monitor de glucosa en el momento de la lectura. Durante 4 meses una muestra de 15 niños (de 9 a 15 años) con Diabetes Mellitus tipo 1, utilizaron el prototipo 3 veces al día, mostrando que el sistema se integra fácilmente a la vida cotidiana, y los padres valoran el sentido de tranquilidad que les ofrece el sistema.

Logan (2007) et al. En [31] desarrolló un sistema de gestión de la diabetes, el sistema consta de los componentes del paciente, un sistema de deposito de datos y apoyo a la decisión, presentación de informes médicos y alertas. El teléfono recibe las lecturas del automonitoreo por medio de bluetooth posteriormente estos datos son transmitidos de forma segura al servidor de datos. Los pacientes tienen la opción de revisar las lecturas previas que se almacenan en el teléfono en forma de tablas, gráficos y formatos de resumen. El uso interactivo de este sistema entre los pacientes y médicos pueden iniciar una solicitud de informe de fax que se enviará al número de fax pre

programado de la oficina de los médicos, este informe consta de la lecturas del paciente de los últimos 30 dias. La evaluación se realizo mediante 2 fases: la primera incluyó una serie de reuniones de grupos focales con los pacientes y proveedores de atención primaria y la segunda se utilizaron 33 pacientes diabéticos con hipertensión no controlada, en un tiempo 4 meses. Demostrando una reducción significativa de la presión arterial reduciendo en 11.5 (+/- SD 13/7)mm Hg.

Capítulo III

3.1 DESCRIPCIÓN DEL TRABAJO.

En el presente trabajo se propone un sistema multiagente, para que un paciente con DM2 sea capaz de regular su presión arterial y glucosa. El sistema está formado por tres agentes principales. La figura 1 muestra la manera en que interactúan los agentes. El Agente Interfaz es el agente principal encargado de recaudar y proporcionar la información necesaria a los dos agentes que llevan a cabo los cálculos de la recomendación alimenticia y la evaluación de dicha recomendación. En el caso del Agente Dieta el agente principal le proporciona los datos antropométricos y preferencias alimenticias necesarias para calcular un plan alimenticio para que un paciente con DM2 sea capaz de regular su glucosa y presión arterial. Por otro lado el Agente Evaluación recibe los datos del automonitoreo de glucosa y presión arterial, antes y después de la ingesta nutricional, ya que la diferencia de estos valores refleja la efectividad de la recomendación.

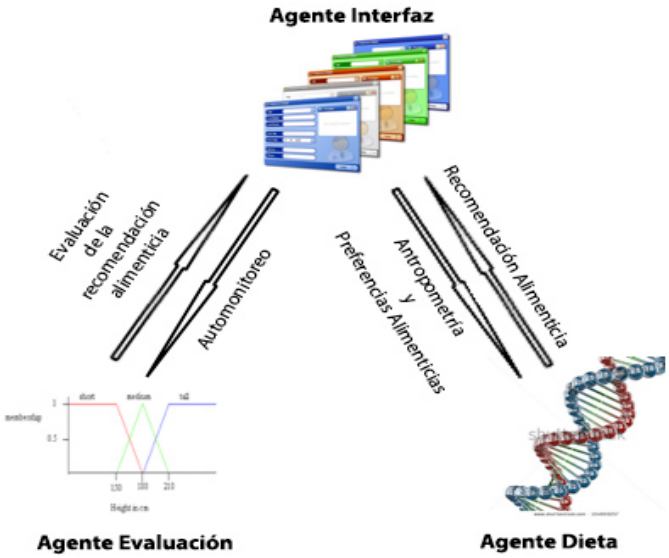


Figura 1. Modelo del sistema multiagente.

3.2 AGENTE INTERFAZ.

El primer agente contiene una interfaz dividida en tres secciones: Usuario, Monitoreo y Preferencias Alimenticias. Este agente lleva a cabo la tarea de recaudar la información necesaria para conocer el estado físico del paciente, creando un expediente personal y recaudar la información alimenticia dependiendo los gustos del paciente, este

expediente es almacenado en la base de datos mostrada en la figura 2 para su posterior utilización.

3.2.1 Expediente Personal.

El expediente personal contiene la información necesaria para conocer el estado físico y bioquímico de un paciente con DM2, esta información se almacena en la base de datos mostrada en la figura 2, para posteriormente poder analizarla. Los datos que forman el expediente personal están divididos en 3 tipos: datos personales, datos clínicos y datos de automonitoreo.

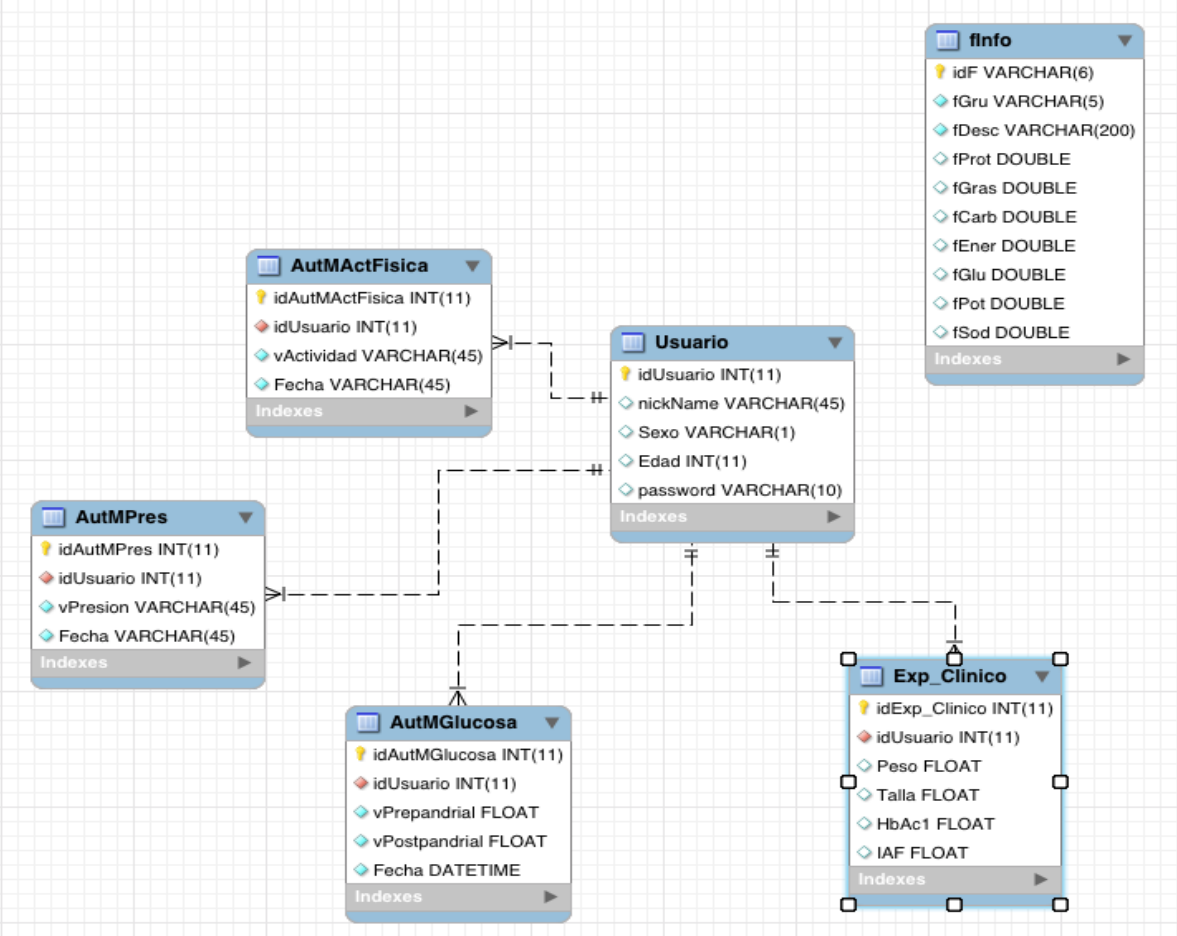


Figura 2. Base de datos del expediente personal.

Como datos personales se consideraron los siguientes:

- ID (Identificador de usuario).
- Nombre.
- Edad.

- Sexo.

Los datos clínicos son los siguientes:

- Peso.
- Talla.
- HbAc1(Hemoglobina glicosilada).
- IAF(Índice de actividad física).

Datos de automonitoreo:

- Glucemia, pre y postprandial.
- Presión arterial, pre y postprandial.
- Actividad física.

3.2.2 Evaluación antropométrica.

La evaluación antropométrica es un conjunto de mediciones corporales en las que se determinan los diferentes grados de nutrición de un individuo mediante parámetros antropométricos e índices derivados de la relación entre los mismos.

- Peso. Esta medición se realiza colocando al paciente sobre una báscula, que conviene que precise en fracciones de 10 gramos [42].
- Talla. El paciente se coloca de pie, erguido y con la espalda colocada en forma vertical del aparato medidor. La precisión debe ser, al menos, de fracciones de 10 milímetros [42].

Las variables anteriores son de almacenadas en una base de datos. Después de recaudar esta información el agente, en su segunda sección, es capaz de realizar una evaluación antropométrica respecto a ellos, calculando su IMC mediante la siguiente ecuación:

$$Imc = \frac{P}{T^2}$$

Donde:

$Imc = Indice\ de\ masa\ corporal.$

$P = Peso\ (kg).$

$T\ = Talla\ (mts).$

y su peso ideal de la siguiente forma:

$$Pi_M = T^2 * 23 \, , Pi_F = T^2 * 21.5$$

Donde:

$Pi_M = Peso\ ideal\ masculino.$

$Pi_F = Peso\ ideal\ femenino.$

$T = Talla\ (mts).$

3.2.3 Evaluación bioquímica.

La evaluación bioquímica son los indicadores bioquímicos que se evalúan en muestras de sangre, orina y heces fecales, nos proporcionan información sobre concentraciones plasmáticas de glucosa y presión arterial. En este caso se realiza cumpliendo los siguientes parámetros proporcionados por la ADA(American Diabetes Association) [32] mostrados en las tablas 2, 3 y 4.

	Prepandial	Postprandial	Antes de Acostarse
Glucemia (mg/dl)	90-130 mg/dl	<=180 mg/dl	< 140 mg/dl

Tabla 2. Índices de Glucemia.

	Optimo	Aceptable	Comprometido
Presión arterial (mmHg)	<135/85	<=150/90	>150/90

Tabla 3. Índices de Presión Arterial.

	Ligero	Moderado	Fuerte
Actividad Física	<10 min. 1-3 días. Índice: 1.375	10-20 min 3-5 días. Índice: 1.55	30-40 min 6-7 días. Índice:1.725

Tabla 4. Índices Aceptables de Actividad Física.

Si alguno de estos parámetros, está en un rango peligroso, el agente es capaz de emitir una alerta en pantalla para el usuario.

3.2.4 Requerimientos Nutricionales.

Antes del cálculo de la recomendación alimenticia que es el aspecto principal de este sistema, se deben calcular ciertos parámetros como el RCT (requerimiento calórico total), estos son necesarios para proporcionar al paciente los nutrientes apropiados para mantenerlo en un peso ideal. Para ello es necesario utilizar la ecuación de Harris-

Benedict, debido a que considera el factor de actividad física, considerado en este proyecto [33]:

$$RCT_M = 10 * P + 6.25 * T - 5 * E + 5$$
$$RCT_F = 10 * P + 6.25 * T - 5 * E - 161$$

Donde:

$RCT_M = Requerimiento\ calórico\ total\ masculino.$

$RCT_F = Requerimiento\ calórico\ total\ femenino.$

$P = Peso\ (kg).$

$T = Talla\ (mts).$

$E = Edad.$

A continuación se multiplíca el RCT por el IAF para obtener el RCTF:

$$RCTF_{M,F} = RCT_{M,F} * IAF$$

Donde:

$RCTF_{M,F} = Requerimiento\ calórico\ total\ final\ masculino\ o\ femenino.$

$RCT_{M,F} = Requerimiento\ calórico\ total\ masculino\ o\ femenino.$

$IAF = Indice\ de\ actividad\ física.$

El índice de actividad física se obtiene del automonitoreo, dependiendo de la cantidad de actividad realizada se asigna el valor de acuerdo a los parámetros proporcionados por la ADA(American Diabetes Association) [32].

A partir de estos datos es posible calcular, el gasto energía en este caso particular para un paciente con diabetes tipo 2, la cantidad de carbohidratos, proteínas, grasas y sodio es calculada por las siguientes ecuaciones:

$$Carb = (RCTF_{M,F} * 0.5)/4$$
$$Prot = (RCTF_{M,F} * 0.3)/9$$
$$Gras = (RCTF_{M,F} * 0.2)/4$$
$$Sodm = RCTF_{M,F}/1000$$

Donde:

$Carb = Carbohidratos\ (g).$

$Prot = Proteinas\ (g).$

$Gras = Grasas\ (g).$

$Sodm = Sodio\ (mg).$

$$RCTF_{M,F} = Requerimiento\ calórico\ total\ final\ masculino\ o\ femenino.$$

Un plan de alimentación para la diabetes es una guía que le dice al paciente qué tipos de alimentos debe comer y en qué cantidad durante las comidas y como colaciones. Un buen plan de alimentación debe amoldarse a su horario y hábitos de alimentación. El plan adecuado de alimentación ayuda a controlar mejor su nivel de glucosa en la sangre, presión arterial y colesterol, además de mantener el peso apropiado. Si debe bajar de peso o mantener su peso actual, su plan de alimentación puede ayudarlo.

Las personas con diabetes deben prestar particular atención para asegurarse de que exista un equilibrio entre los nutrientes de sus alimentos para ayudar a controlar su nivel de glucosa.

Hidratos de carbono. En pacientes con DM2 el requerimiento mínimo diario es de 50g para evitar la cetosis condicionada por el catabolismo proteico y graso. Son preferibles los carbohidratos complejos, que tienen la característica de absorberse lentamente debido la liberación gradual al torrente circulatorio de la glucosa que contienen, por lo que ejercen una acción moduladora sobre la concentración evitando las bruscas oscilaciones que condicionan hiperglucemia [43].

Proteínas. Para prevenir el daño renal en los diabéticos las proteínas se calculan a 0.8 g/kg de peso corporal/día en lugar de 1 g/kg/día. En los que ya tienen nefropatía la restricción es mayor (0.6 g/kg/día) para reducir la proteinuria y retrasar la progresión hacia insuficiencia renal [43].

Grasas. En los diabéticos se debe reducir a 30% e integrarse fundamentalmente por grasas insaturadas para reducir la ingesta de colesterol a cifras menores de 300 mg por día y disminuir el riesgo de aterogénesis [43].

Sodio. La cantidad de sal debe reducirse a 3 g/día debido a que en el diabético la hiperinsulinemia condicionada por la enfermedad incrementa la reabsorción renal de Na+ y de forma alterna estimula el sistema simpático, lo que favorece la asociación de hipertensión arterial y diabetes. En el paciente hipertenso se recomienda un consumo no mayor de 2.4 g/día [43].

3.2.5 Preferencias Alimenticias.

La tercera sección, es donde el usuario selecciona su preferencia alimenticia, esto se lleva a cabo proporcionando una lista de alimentos proporcionados por la USDA National Nutrient Database ver. 27 [32], cuenta con un catálogo de 8 618 diferentes alimentos descritos por su índice nutrimental, o si lo desea, el usuario puede agregar algún alimento de su preferencia , si conoce su información nutricional, para ser almacenado. Cada uno de ellos contiene la información nutrimental necesaria para realizar una recomendación alimenticia adecuada al paciente. En este caso se consideraron 7 aspectos: proteínas (g), carbohidratos (g), grasas (g), energía (kcal), índice glucémico (mg), potasio (mg), sodio (mg). Cada alimento esta descrito por esta información nutrimental. El usuario al considerar un alimento necesita ingresar la cantidad de gramos en el caso de alimentos solidos o tazas en líquidos a ingerir, a partir de esto se calcula la información nutrimental total del alimento.

Estos alimentos se presentan al usuario en 25 categorías formando un árbol de la siguiente manera:

- Dairy and Egg Products (Lácteos y Huevos).
- Spices and Herbs (Especias y Hierbas).
- Baby Foods (Alimentos para bebés).
- Fats and Oils (Grasas y aceites).
- Poultry Products (Productos avícolas).
- Soup, Sauces and Gravies (Sopas y Salsas).
- Sausages and Luncheon Meats (Embutidos y Carnes de almuerzo).
- Breakfast Cereals (Cereales de desayuno).
- Fruits and Fruit Juices (Frutas y jugos de frutas).
- Pork Products (Productos de cerdo).
- Vegetables and Vegetable Products (Verduras y Hortalizas).
- Nut and Seed Products (Productos de Nueces y Semillas).
- Beef Products (Productos de Res).
- Beverages (Bebidas).
- Finfish and Shellfish Products (Pescados y mariscos).
- Legumes and Legume Products (Productos de Legumbres y Leguminosas).
- Lamb, Veal and Game Products (Productos de Cordero, Ternera y Caza).

- Baked Products (Productos Horneados).
- Sweets (Dulces).
- Cereal Grains and Pasta (Granos de Cereal y Pastas).
- Fast Foods (Comidas Rápidas).
- Meals, Entriees and Sidedishes (Colaciones, Entradas y Guarniciones).
- Snacks (Bocadillos).
- Ethnic Foods (Comidas Etnicas).
- Restaurant Foods (Comidas de Restaurant).

El índice glucémico, potasio y sodio son tres aspectos importantes que se consideraron debido a que la recomendación alimenticia esta orientada a pacientes con DM2. El índice glucémico es un sistema para cuantificar la respuesta glucémica de un alimento que contiene la misma cantidad de carbohidratos que un alimento de referencia. La mayor parte de los alimentos contienen carbohidratos, lo importante, a parte de la cantidad es necesario saber que tan rápido se digieren y absorben. En el caso de la diabetes se deben controlar estos niveles glucémicos. El sodio y el potasio son los reguladores nutrimentales de la presión arterial, el exceso de sodio y el déficit de potasio de nuestra alimentación, son características en el desarrollo de la hipertensión [34].

3.3 AGENTE DIETA

El segundo agente es el encargado del cálculo de la recomendación alimenticia que ayudara al paciente a regular sus índices de glucosa y presión arterial. Este agente toma como aspectos principales, los datos provenientes del Agente Interfaz, en especifico los siguientes:

- RCTF
- Peso
- Talla
- Total de Carbohidratos
- Total de Proteínas
- Total de Lípidos
- Total de Sodio
- Total de Potasio

- Total de Índice Glucémico
- Información alimenticia(idF, fDesc, fCarb, fProt, fGras, fEner, fGlu, fPot, fSod)

Los datos anteriores son utilizados por el algoritmo genético que es el encargado de generar la recomendación alimenticia. El algoritmo genético es una técnica de selección combinatoria basado en el proceso genético de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acuerdo a los principios de la selección natural y la supervivencia de los más fuertes, postulados por Darwin. Por imitación de este proceso, los Algoritmos Genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas [40].

3.3.1 Algoritmo Genético.

Un algoritmo genético consiste en una función matemática (funcion fitness) o una rutina de software que toma como entradas a los ejemplares (cromosomas) y retorna como salidas (genes) cuales de ellos deben generar descendencia para la nueva generación.

El corazón del algoritmo genético es el cromosoma. El cromosoma representa una posible solución y se divide en múltiples genes. En este caso un gen esta representado por un alimento descrito por un conjunto de nutrientes y un cromosoma como un conjunto de alimentos que representan un posible plan alimenticio para el paciente. Para la creación de los cromosomas en particular se consideraron los siguientes dos espacios vectoriales:

$$Al(Alimentos)$$

$$Nu(Nutrientes)$$

Donde:

$$Al = (a_1, a_2, a_3, \dots, a_x)$$

$$Nu = (n_1, n_2, n_3, \dots, n_y)$$

Siendo:

$x = número\ de\ alimentos.$

$y = número\ de\ nutrientes.$

Obteniendo el conjunto de genes representados por la variable Dis :

$$Dis = \{(a_i, n_{j1}, \dots, n_{jx}), a_i \in Al(n_{j1}, \dots, n_{jx}) \in Nu\}$$

Definiendo que el conjunto $(n_{j1}, \dots, n_{jx})$ estará definido por nu:

$$Dis = \{(a_i, nu_j), a_i \in Al(nu_j) \in Nu\}$$

Quedando la estructura de esta manera para la representación de un alimento:

$$Dis = \{(a_1, nu_1), (a_2, nu_2), (a_3, nu_3), \dots, (a_{z-1}, nu_{z-1})\}$$
$$(a_1, nu_1)\ representa\ un\ alimento.$$

A partir de esto definimos que un recurso, F_x, es asignado a un gen de cromosoma Dis_k de la siguiente manera:

$$Dis(a_i, nu_{j1}, \dots, nu_{jy}) = F_x$$

Cromosoma:

F_1	F_2	F_3	…	F_{n-1}	F_n	X
0	1	2	…	$t-1$	t	$Valor\ de\ aptitud$

Tabla 5. Cromosoma de ejemplo.

Para realizar la evaluación se modifica el fitness puro para los problemas de maximización y minimización:

$$r(i,t) = \sum_{j=1}^{N_c} |s(i,j) - c(i,j)|$$

Donde:

$s(i,j) = valor\ deseado\ para\ el\ intervalo\ \ i\ en\ el\ caso\ j.$

$c(i,j) = valor\ obtenido\ para\ el\ intervalo\ \ i\ en\ el\ caso\ j.$

$N_c = Número\ de\ casos.$

$$s(i,j) = \begin{cases} r(i,t) & minimización. \\ r_{max} - r(i,t) & maximización. \end{cases}$$

Para determinar una recomendación alimenticia adecuada para un paciente con diabetes mellitus tipo 2 se emplea la siguiente ecuación:

$$Eval(Dis_q) = \sum_{s=1}^{T_c} \left[\left(rG_{max} - (T_{glu(s)} - \partial_{(s)}) \right) + \alpha_{(s)} + \delta_{(s)} \right]$$

Donde:

$\partial_{(s)} = Glucemia\ obtenida\ para\ el\ caso\ (s).$

$T_{\mathrm{glu(s)}} = Glucemia\ esperada\ para\ el\ caso\ (s).$

$\alpha_{(s)} = Función\ aptitud\ para\ la\ presión\ arterial.$

$\delta_{(s)} = Función\ aptitud\ para\ los\ nutrientes.$

$rG_{max} = Cota\ superior\ glucemica\ del\ fitness\ puro.$

$T_c = Tamaño\ del\ cromosoma.$

Para la estabilizar de presión arterial se utilizara la siguiente ecuación:

$$\alpha_{(s)} = rP_{max} - \left(R_{pre(s)} - \theta_{(s)} \right)$$

Donde:

$\alpha_{(s)} = Función\ \ aptitud\ para\ la\ presión\ arterial.$

$R_{pre(s)} = Glucemia\ esperada\ para\ el\ caso\ (s).$

$\theta_{(s)} = presión\ arterial\ obtenida\ para\ el\ caso\ (s)\ .$

$rP_{max} = Cota\ superior\ presión\ arterial\ del\ fitness\ puro.$

Para la conseguir un aporte nutricional adecuado se utilizara la ecuación:

$$\delta_{(s)} = \sum_{i=1}^{T_n} \left[rN_{max} - (C_{nut(i)} - \varphi_{(i)}) \right]$$

Donde:

$\delta_{(s)} = Función\ \ aptitud\ para\ los\ nutrientes.$

$C_{nut(s)} = Aporte\ nutricional\ \ esperado\ para\ el\ caso\ i.$

$\varphi_{(s)} = Aporte\ nutricional\ obtenido\ para\ el\ caso\ i.$

$rN_{max} = Cota\ superior\ nutricional\ del\ fitness\ puro.$

$T_n = Tamaño\ de\ los\ nutrientes\ del\ recurso\ para\ el\ caso\ n.$

Reemplazando las ecuaciones obtenemos la función de aptitud del algoritmo genético:

$$Eval(Dis_q) = \sum_{s=1}^{T_c} \left[rG_{max} - (T_{glu(s)} - \partial_{(s)}) + rP_{max} - \left(R_{pre(s)} - \theta_{(s)} \right) + \sum_{i=1}^{T_n} \left[rN_{max} - (C_{nut(i)} - \varphi_{(i)}) \right] \right]$$

A partir de esta función debemos obtener la longitud del cromosoma para representar la solución dependiendo la precisión deseada.

$$\max(T_{glu}, R_{pre}, C_{nut}) = \sum_{s=1}^{T_c}\left[rG_{max} - (T_{glu(s)} - \partial_{(s)}) + rP_{max} - \left(R_{pre(s)} - \theta_{(s)}\right) + \sum_{i=1}^{T_n}[rN_{max} - (C_{nut(i)} - \varphi_{(i)})]\right]$$

$$90 \leq T_{glu} \leq 180$$

$$135.85 \leq R_{pre} \leq 150.90$$

$$C_{nut} \leq RCTF$$

Para el cálculo de los bits requeridos para representar el cromosoma se utiliza la formula:

$$2^{m_j - 1} < \left(b_j - a_j\right) * 10^n \leq 2^{m_j} - 1$$

Donde:

$\left(b_j - a_j\right) = es\ el\ domino\ de\ la\ variable.$

$n = es\ la\ precisión\ requerida.$

$m_j = cantidad\ de\ bits\ requeridos.$

Entonces calculamos la longitud de la cadena para las variables $T_{glu}, R_{pre}, C_{nut}$con una precisión de 2 dígitos después del punto decimal:

$$T_{glu} = (180 - (90) * 100) = 9000 \ \rightarrow 2^{14} < 9000 < 2^{15} \rightarrow m_1 = 15$$

$$R_{pre} \ = (150.90 - (135.85) * 100) = 1505 \rightarrow 2^{11} < 1505 < 2^{12} \rightarrow m_2 = 12$$

$$C_{nut} = (2500) \rightarrow 2^{12} < 2500 < 2^{13} \rightarrow m_3 = 13$$

Longitud total del cromosoma: $m_1 + m_2 + m_3 = 40$.

En particular, se tomaron como tamaño de la población 377 individuos, y 610 evoluciones permitidas, la razón de esto es que el tiempo de respuesta del algoritmo es de 60s, lo cual es más cómodo para el usuario.

Para la selección a partir de la población inicial se procede a ordenar los individuos de acuerdo a su aptitud, escogiendo de manera aleatoria un número n de individuos (generalmente n es igual a 3) para realizar un torneo entre ellos, donde el más apto es seleccionado para formar la población intermedia, este procedimiento se repite hasta

completar el número N de individuos de la población inicial. La población intermedia obedece su nombre porque esta en la transición entre una población i y la población $i + 1$.

A esta población intermedia se le aplican los operadores clásicos para explorar nuevos espacios de búsqueda y calidad, siendo los más citados “Crossover” o cruzamiento y Mutación.

El crossover o cruza se realiza usando 1 punto, donde el cromosoma hijo es generado a partir del cromosoma de dos padres como lo muestra la figura 3:

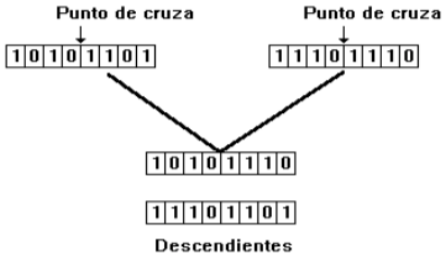


Figura 3. Ejemplo de crossover del algoritmo genético.

La mutación se realiza mediante la técnica de Uniform Mutation (Mutación uniforme), debido a que la distribución de probabilidad usada es uniforme, en este caso es de 5% debido a que los valores más comúnmente utilizados para la muta son de bajo porcentaje, van del 1 al 5% [41].

3.4 AGENTE EVALUACIÓN.

El Agente Evaluación es el encargado de verificar la efectividad de la recomendación alimenticia proporcionada al usuario por el Agente Dieta, evaluándola mediante un control difuso. Esto se realiza verificando el factor de regulación generado por los automonitoreos de glucosa y presión arterial antes y después de la ingesta alimentaria. Dependiendo del valor de este factor de regulación, el control da un veredicto de que tan efectiva fue la recomendación propuesta.

Para el cálculo del factor de regulación de la glucosa se utiliza la siguiente formula:

$$FacReg_{Glu} = Glucosa_{pre} - Glucosa_{post}$$

Donde:

$FacReg_{Glu} = Factor\ de\ regulación\ de\ Glucosa.$

$Glucosa_{pre} = Automonitoreo\ de\ Glucosa\ prepandrial.$

$Glucosa_{post} = Automonitoreo\ de\ Glucosa\ postpandrial.$

Y para el cálculo del factor de regulación de la presión arterial se utiliza la formula:

$$FacReg_{Pre} = Presión_{pre} - Presión_{post}$$

Donde:

$FacReg_{Pre} = Factor\ de\ regulación\ de\ Presión\ arterial.$

$Presión_{pre} = Automonitoreo\ de\ Presión\ arterial\ prepandrial.$

$Presión_{post} = Automonitoreo\ de\ Presión\ arterial\ postpandrial.$

El valor de la evaluación se obtiene mediante la suma de las dos variables anteriores, calculada por la siguiente formula:

$$Eval = FacReg_{Glu} + FacReg_{Pre}$$

Donde:

$Eval = Factor\ de\ evaluación\ de\ la\ recomendación\ alimenticia\ propuesta.$

$FacReg_{Glu} = Factor\ de\ regulación\ de\ Glucosa.$

$FacReg_{Pre} = Factor\ de\ regulación\ de\ Presión\ arterial.$

El control difuso pretende proporcionar la evaluación de una recomendación alimenticia dada a partir de los valores anteriores $FacReg_{Glu}\ y\ FacReg_{Pre}$, con la finalidad de saber que tan efectivo resulto su consumo.

- Se cuenta con el factor de regulación de glucosa con un rango de valores de -120 a 120.

- Se cuenta con el factor de regulación de presión arterial con un rango de valores de -80 a 80.
- La evaluación se puede verificar mediante valores con un rango de -200 a 200.

3.4.1 Variables de estado.

- Factor de regulación de Glucosa

  Se asumen 5 etiquetas lingüísticas como se muestra en la figura 4:

  - Muy mala(Mm), descrita por una función trapezoidal con valores de -160, -120,-80,60.
  - Buena(B), descrita por una función triangular con valores de -90,-45,0.
  - Normal(N), descrita por una función triangular con valores de -50,0,50.
  - Muy buena(MB), descrita por una función triangular con valores de 0,45,90.
  - Mala(M), descrita por una función trapezoidal con valores de 60, 80,120,160.

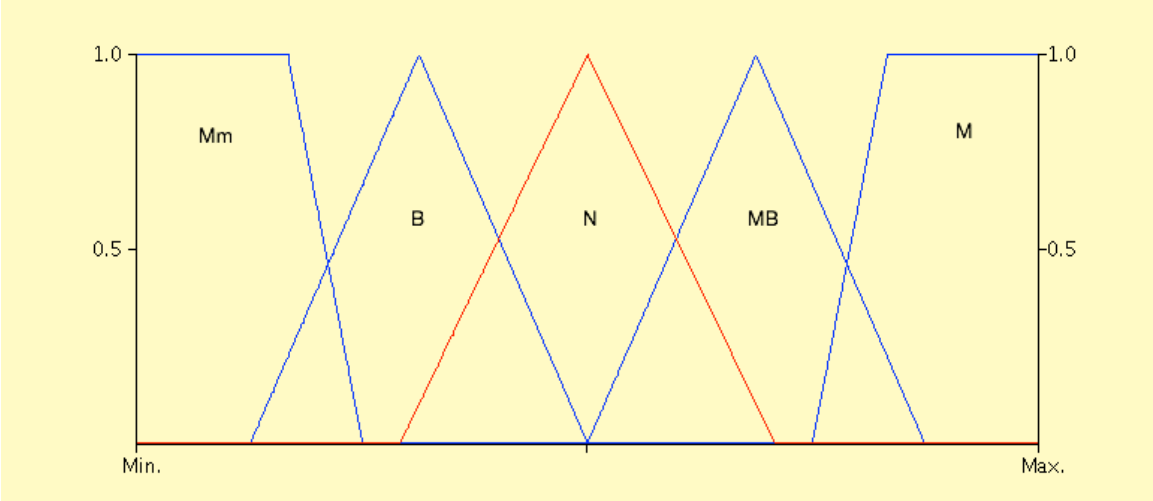


Figura 4. Funciones de membresía del factor de regulación de glucosa.

- Factor de regulación de Presión arterial

  Se asumen 5 etiquetas lingüísticas como se muestra en la figura 5:

  - Mala(M), descrita por una función trapezoidal con valores de -106, -80,--53.3,-40.
  - Buena(B), descrita por una función triangular con valores de -50,-25,0.
  - Normal(N), descrita por una función triangular con valores de -20,0,20.
  - Muy buena(MB), descrita por una función triangular con valores de 0,25,50.

- Muy mala(Mm), descrita por una función trapezoidal con valores de 40, 53.3,80,106.

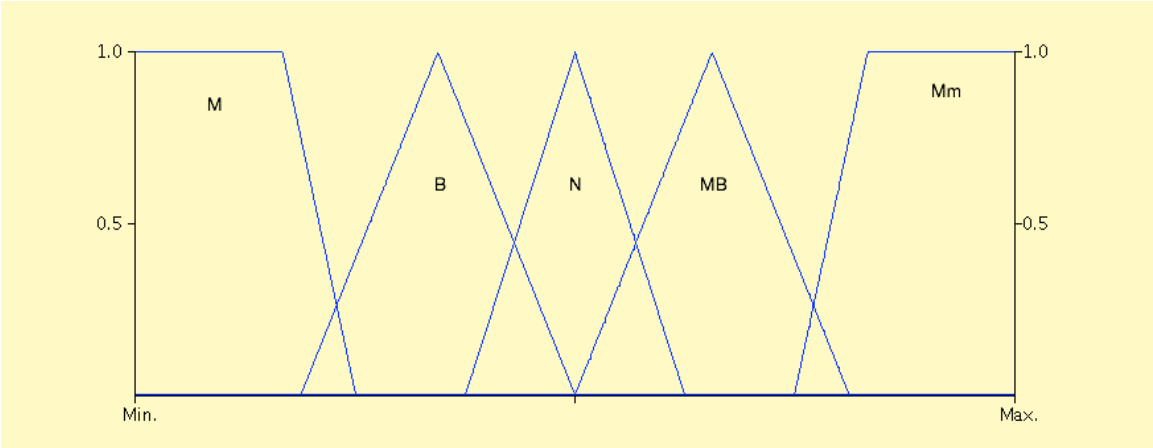


Figura 5. Funciones de membresía del factor de regulación de presión arterial.

3.4.2 Variables de control.

- Evaluación

  Se asumen 5 etiquetas lingüísticas como muestra la figura 6:

  - Muy mala(Mm), descrita por una función trapezoidal con valores de -266, -200,-133.3,-66.6.
  - Buena(B), descrita por una función triangular con valores de -133.3, -66.6, 0.
  - Normal(N), descrita por una función triangular con valores de -66.6, 0, 66.6.
  - Muy buena(MB), descrita por una función triangular con valores de 0, 66.6, 133.3.
  - Mala(M), descrita por una función trapezoidal con valores de 66.6, 133.3, 200, 266.

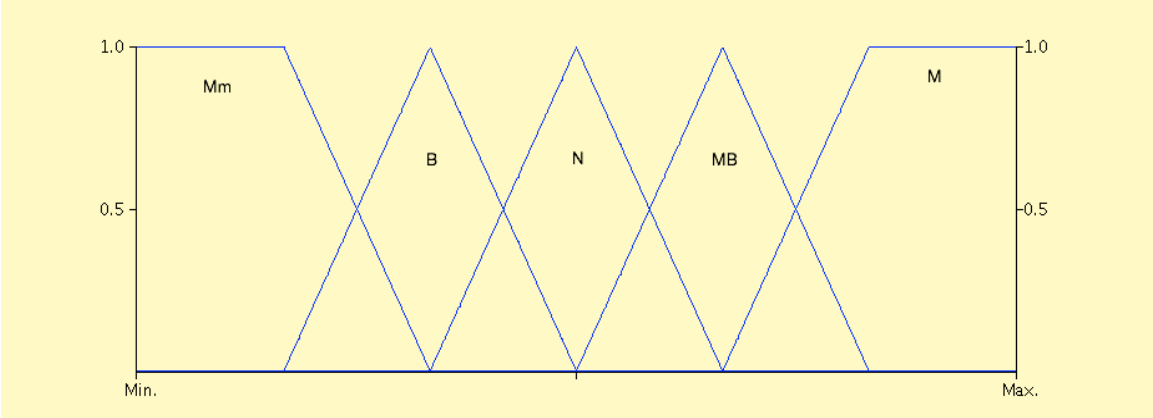


Figura 6. Funciones de membresía de la evaluación.

3.4.3 Reglas Difusas.

La figura 7 muestra la FAM (Fuzzy Asocciation Matrix) considerada para la variable de control Evaluación.

FacRegPre

FacRegGlu	mala	buena	normal	muybuena	muymala
muyMala	mala	mala	mala	mala	muymala
buena	mala	buena	buena	muybuena	mala
normal	mala	buena	normal	buena	mala
muybuena	normal	muybuena	muybuena	muybuena	normal
mala	mala	normal	normal	normal	muymala

Figura 7. Fuzzy Asocciation Matrix de la evaluación.

3.4.4 Parámetros de la inferencia difusa.

Se utilizaron los siguientes operadores:

- Conectiva AND (^): mínimo.
- Conectiva OR (*vee*): máximo.
- Implicación difusa (→): Mamdani.
- Modus ponens difuso: min-max (composición de conj. difuso con relación difusa)
  Agregación de las salidas difusas de las reglas activadas: OR (máximo)
- Operador de fuzzyficación: singleton.
- Operador de desfuzzificación: centro de masas.

3.5 Comunicación entre agentes

Una característica de los sistemas multiagente es que los agentes individuales se comunican e interactúan. Esto se logra mediante el intercambio de mensajes, siempre y cuando los agentes están de acuerdo sobre el formato y la semántica de estos mensajes. Los estándares FIPA ayudan a los agentes a interactuar con agentes escritos en otros lenguajes y que se ejecutan en otras plataformas.

En particular, la comunicación entre los agentes de este trabajo se realiza siguiendo los estandares de FIPA en la plataforma Jade, usando mensajes FIPA-ACL que nos permite transmitir una serie de conocimento entre ellos como muestra la figura 8:

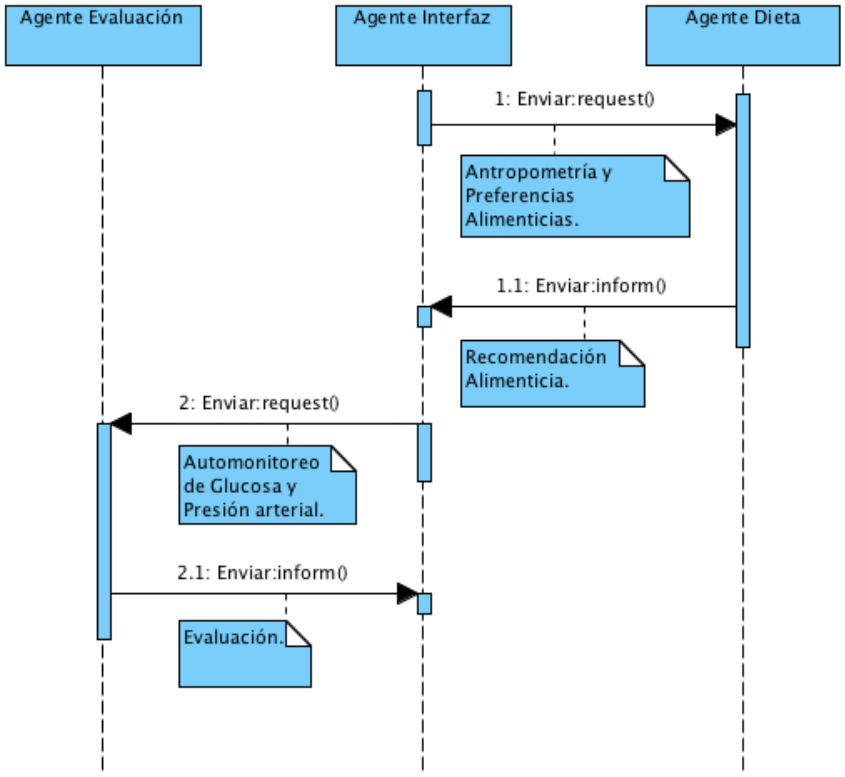


Figura 8. Diagrama de comunicación del sistema multiagente.

Se utilizaron dos tipos de mensajes principales para la comunicación entre los agentes en el sistema, en este caso fueron de tipo *request* e *inform*.

Donde:

- Request. Es donde el agente emisor solicita al agente receptor que realice una acción.

- Inform. Es donde se informa los resultados de la solicitud.

La figura demuestra el proceso que lleva a cabo el sistema multiagente mediante el paso de mensajes FIPA-ACL. El usuario es el encargado de proporcionar al Agente Interfaz la información necesaria para crear su expediente personal, que contiene los datos necesarios para que se realice la evaluación antropométrica explicada en la sección 3.2.1, también el usuario debe introducir su automonitoreo de glucosa y presión arterial preprandial para la evaluación bioquímica y por último sus preferencias alimenticias, a partir de haber proporcionado al Agente Interfaz los datos anteriores el usuario es capaz de invocar el evento número 1 del proceso cuando el Agente Interfaz teniendo la información anterior que es necesaria para la elaboración de una recomendación alimenticia, envía un mensaje de tipo Request proporcionando en el contenido del mensaje los valores explicados en las secciones 3.2.1, 3.2.2, 3.2.4, el Agente Dieta recibe los datos y elabora el plan alimenticio mediante un algoritmo genético devolviendo esta información automáticamente en el evento 1.1 con un mensaje de tipo Inform, indicando las cantidades necesarias que el usuario necesita consumir para completar la ingesta nutricional adecuada.

Por otro lado después de la ingesta alimentaria el usuario proporciona al Agente Interfaz el automonitoreo de la glucosa y presión arterial postprandial, el evento 2 es invocado por el usuario para enviar un mensaje de tipo Request al Agente Evaluación conteniendo los valores indicados en la sección 3.4. A partir de la recepción del mensaje, el Agente Evaluación mediante un control difuso muestra que tan efectiva fue la recomendación. En el evento 2.1 el Agente Evaluación devuelve el valor que indica que tan efectiva fue la recomendación alimenticia automáticamente por medio de un mensaje de tipo Inform completando el ciclo de comunicación del sistema multiagente.

Capítulo IV

En esta sección se explica el procedimiento y la descripción de las actividades realizadas, está dividida en 3 secciones pertenecientes a cada uno de los agentes que componen el sistema.

El sistema multiagente está implementado bajo el lenguaje Java 1.8 usando la plataforma Jade (Java agent develoment framework) que simplifica la implementación de sistemas multiagente a través de un framework que cumple con las especificaciones FIPA (Foundation for Intelligent Physical Agents).

Como se mencionó en la sección 3.1 el sistema esta formado por 3 agentes: Agente Interfaz, Agente Dieta y Agente Evaluación. El Agente Interfaz, para llevar a cabo la tarea de almacenar los datos del paciente, y crear un expediente personal, se utilizó Mysql 5.26.23. Por otra parte, los Agentes Dieta y Evaluación, usan herramientas adicionales para llevar a cabo sus tareas. Jgap 3.4.4 (Java genetic algoritm package), en el Agente Dieta, es un componente de programación de algoritmos genéticos en Java. Proporciona mecanismos genéticos básicos para aplicar principios evolutivos en la solución de problemas en particular, proporcionar una recomendación alimenticia a un paciente con DM2.

El Agente Evaluación utiliza un control difuso que lleva a cabo la evaluación de la recomendación alimenticia proporcionada. El control difuso esta diseñado mediante la herramienta Xfuzzy 3 e implementado con JFuzzyLogic 2.1 que son entornos de desarrollo para sistemas difusos que integran un conjunto de herramientas que facilitan al usuario para cubrir las etapas del proceso de diseño de los sistemas de inferencia basados en lógica difusa, desde su descripción inicial hasta su ejecución final.

4.1 PROCEDIMIENTO Y DESCRIPCIÓN DE ACTIVIDADES

4.1.1 AGENTE INTERFAZ

El Agente Interfaz encargado de llevar el control de la información del usuario tanto personal como clínico y de automonitoreo también de realizar las evaluaciones antropométricas y bioquímicas de un paciente con DM2, está formado por distintos módulos que componen la interfaz para llevar a cabo estas tareas, en este caso son:

- Acceso
- Panel de Control de Usuario
    - Usuario
    - Monitoreo
- Panel de Control Dieta
    - Alimentos
    - Agregar Alimentos

4.1.1.1 Acceso

La figura 9 muestra la ventana de inicial, esta ventana se encarga de la gestión de los usuarios, desde ella, un usuario puede crearse o acceder al sistema multiagente.

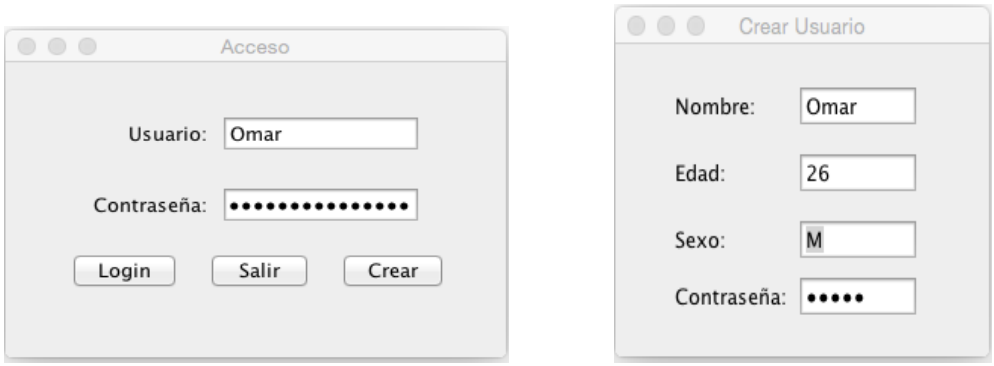


Figura 9. Ventana de Acceso y Crear usuario.

4.1.1.2 Panel de control Usuario

4.1.1.2.1 Usuario

Cuando un usuario ingresa al sistema, accede automáticamente al panel de control de usuario (ver figura 10), esta interfaz esta divida en dos pestañas, Monitoreo y Usuario. En la pestaña de usuario el paciente puede realizar su evaluación antropométrica, la cual le informa su estado físico y requerimientos nutricionales necesarios para su recomendación alimenticia.

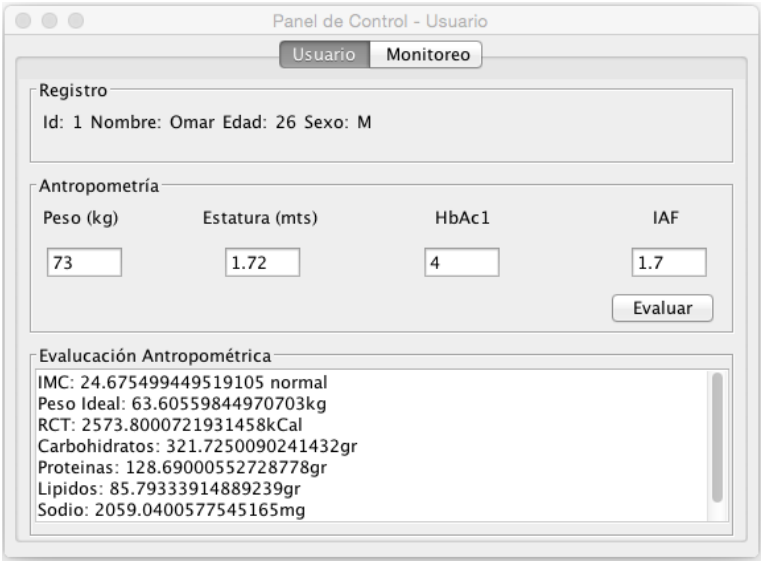


Figura 10. Panel de control usuario.

4.1.1.1.2 Monitoreo

La pestaña Monitoreo (ver figura 11), es la encargada de gestionar el automonitoreo de glucosa, presión arterial y actividad física del paciente con DM2, también la evaluación bioquímica es realizada desde esta sección del panel de control de usuario, desde aquí se emiten las alertas necesarias si es que los automonitoreos de glucosa y presión arterial llegan a ser peligrosos. El automonitoreo se realiza en dos etapas una antes de cada comida (prepandial) y una después de haber ingerido la recomendación alimenticia (postprandial).

Figura 11. Pestaña monitoreo del Panel de control usuario.

Las alertas mostradas en la figura 12, son emitidas si alguno de los valores de automonitoreo de glucosa o presión arterial sobrepasan los parámetros aceptables proporcionados por la ADA(American Diabetes Association) [32]. Es importante mencionar que si alguna de estas alertas es mostrada al usuario debe comunicarse inmediatamente con su médico pues corre peligro de sufrir alguna complicación severa de su padecimiento, en el caso de la glucosa hiperglucemia o hipoglucemia y en la presión arterial sufrir hipertensión.

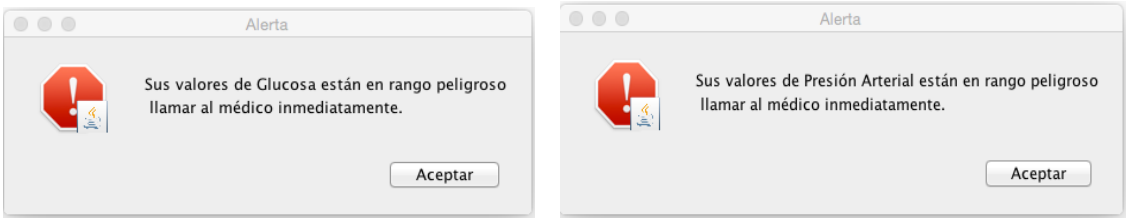


Figura 12. Alertas de Glucosa y Presión arterial.

Despues de que el usuario, ingresa los automonitoreos prepandrial y postprandrial, puede hacer la solicitud pulsando el boton “Agente Evaluación” (ver figura 13) que es el encargado de crear un mensaje de tipo Request al Agente Evaluación mencionado en la sección 3.5, el mensaje creado contiene los factores de regulación mencionados en la seccion 3.4 necesarios para evaluar la recomendación alimenticia por medio de un control difuso.

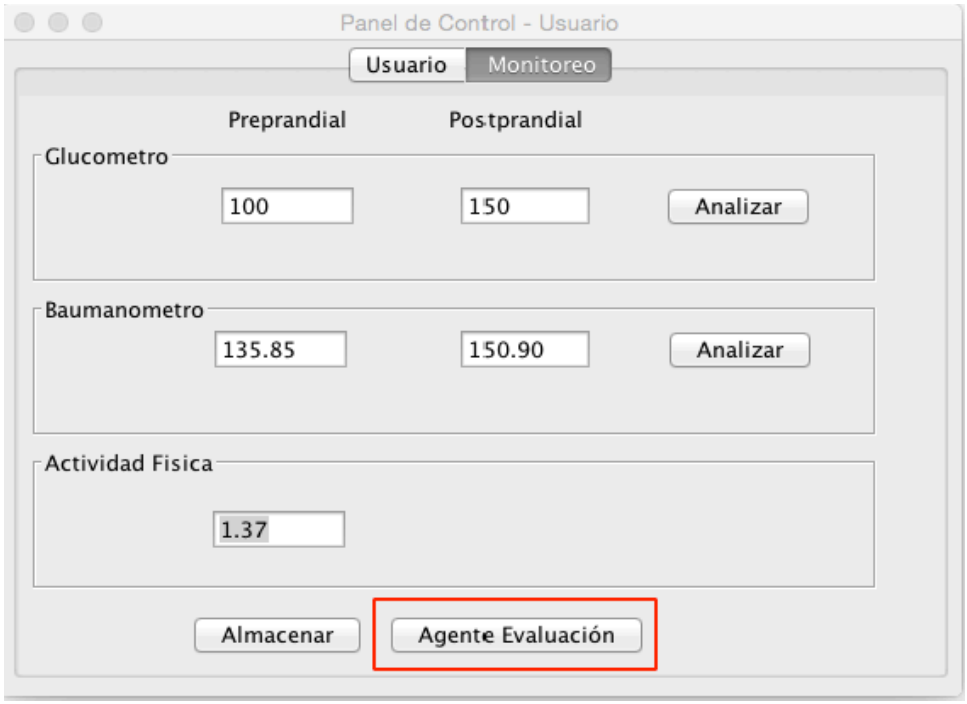


Figura 13. Botón Agente Evalucación

4.1.1.3 Panel de Control Plan Dieta

El Panel de Control Dieta mostrado en la figura 14, es la interfaz encargada de proporcionar al paciente una lista de alimentos junto a su información nutricional. Los alimentos seleccionados son las preferencias alimenticias consideradas al crear la recomendación.

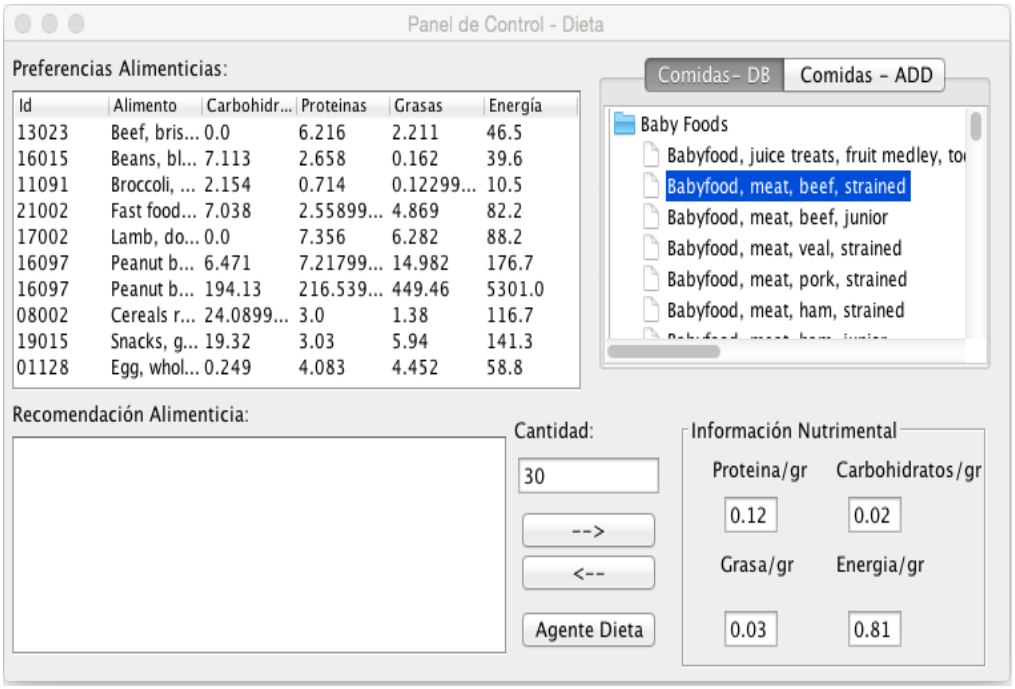


Figura 14. Panel de Control dieta.

Otra de las secciones que forman parte de esta interfaz es Comidas ADD (ver figura 15), que es la encargada de proporcionar al usuario una herramienta que le permita agregar a la base de datos, alimentos que no existan en la base de datos del sistema.

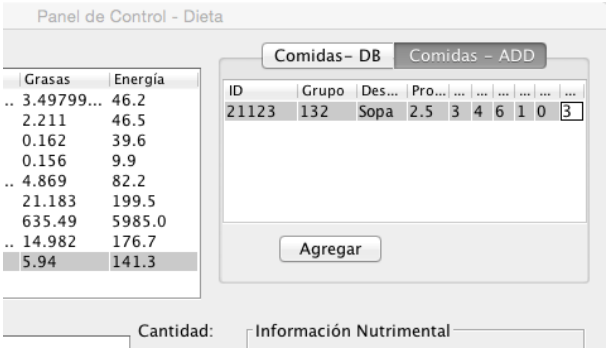


Figura 15. Sección Comidas ADD.

Desde este panel de control, pulsando el botón “Agente Dieta” mostrado en la figura 16, el usuario es capaz de establecer comunicación con el Agente Dieta para que realice la recomendación alimenticia, creando un mensaje de tipo request mencionado en la sección 3.5, conteniendo toda la informacion seleccionada del arbol de alimentos o en su caso de alguno agregado personalmente y los datos pertenecientes a su evaluación antropométrica.

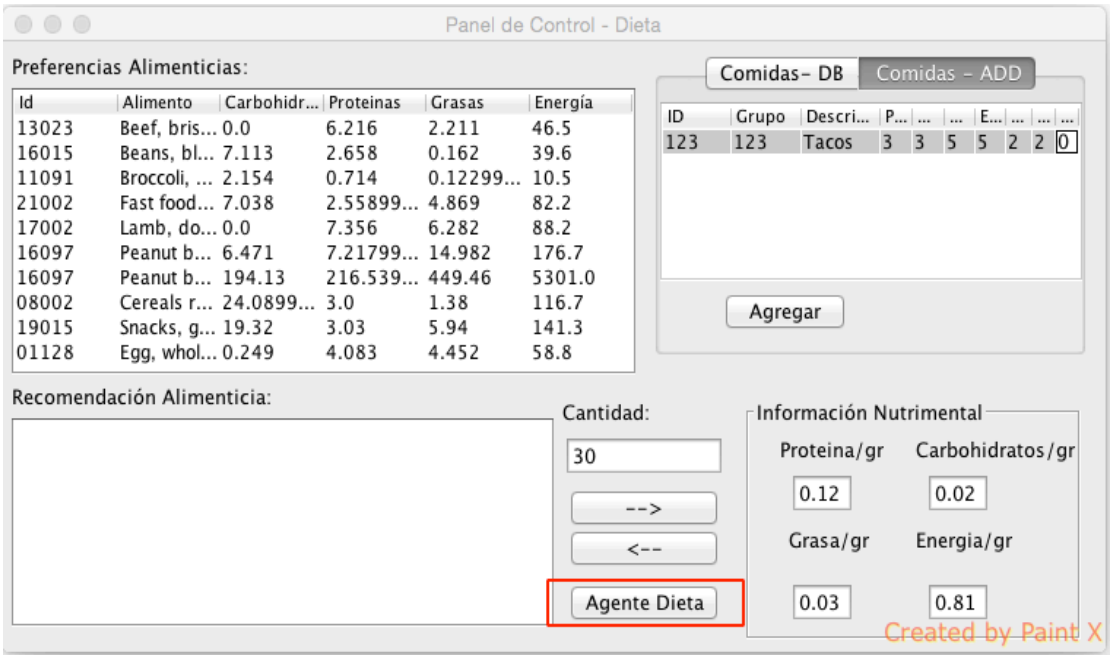


Figura 16. Botón Agente Dieta.

4.1.2 AGENTE DIETA

El Agente Dieta es el encargado de realizar la recomendación alimenticia, no cuenta con una interfaz gráfica para interacuar con el usuario, sin embargo la interaccion es por medio de mensajes FIPA-ACL creados por el usuario conteniendo la información necesaria para la creación de la recomendación alimenticia mencionados en la sección 3.2.2 y 3.2.4. El agente está formado por tres clases, JadeCal, DietCFitnessFunction, DietCMain.

- JadeCal (ver A.1.1). Esta clase es la encargada de establecer la comunicación entre el Agente Interfaz y el Agente Dieta, también se encarga de traducir los mensajes en datos utilizables para crear la recomendación alimenticia.
- DietCMain (ver A.1.3.1). Parte principal del algoritmo genético, esta clase se encarga de crear los cromosomas y genes que componen cada solución posible al problema propuesto.
- DietCFitnessFunction (ver A.1.3.2). Parte del algotimo genético, es la clase encargada de evaluar cada solución hasta encontrar la más apta para ser devuelta al Agente Interfaz en forma de una recomendación alimenticia.

Cuando el algorimo genético encuentra la mejor soluciòn, automaticamente el Agente Dieta crea y envia un mensaje al Agente Interfaz conteniendo la recomendación alimenticia para ser desplegado en su Panel de Control Dieta.

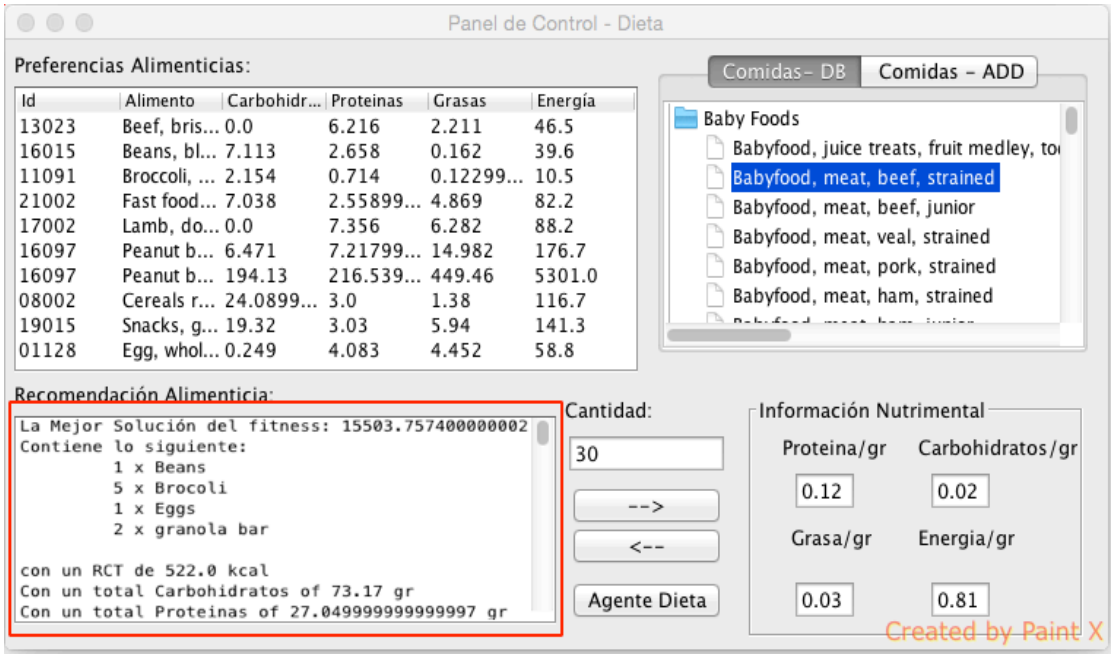


Figura 17. Representación de la recomendación alimenticia.

4.1.3 AGENTE EVALUACIÓN

El Agente Evaluación es el encargado de juzgar que tan efectiva fue la recomendación alimenticia propuesta por el Agente Dieta tomando en cuenta los factores de regulacion mencionados en la sección 3.4. La interacción con el usuario es por medio de mensajes FIPA-ACL que el usuario invoca desde el Agente Interfaz. El agente esta formado por dos clases JadeCalE y ContDifEval.

- JadeCal (ver A.1.1). Esta clase es la encargada de establecer la comunicación entre el Agente Interfaz y el Agente Evaluación, también se encarga de tradcucir los mensajes en datos utilizables para evaluar la recomendación alimenticia.
- ContDifEval (ver A.1.4.1). Esta clase contiene el contiene el control difuso encagado de evaluar la reomendación alimenticia utilizando los factores de regulacion. Automaticamente despues de realizar su tarea informa al Agente interfaz el resultado de la evaluación.

4.2 ANÁLISIS Y RESULTADOS

4.2.1 Análisis de la variable dependiente.

Aplicando 25 pruebas al algoritmo genético propuesto. ¿Existe suficiente evidencia para comprobar que el algoritmo genético es una herramienta capaz de proporcionar un plan alimenticio balanceado para un paciente con DM2?.

- $H_0 =$ El algoritmo genético propuesto no proporciona un plan alimenticio balanceado para un paciente con DM2.
- $H_i =$ El algoritmo genético propuesto es capaz de proporcionar un plan alimenticio balanceado para un paciente con DM2.

Nivel de Significación: Para todo valor de igual o menor que 0.05 se acepta H_i y se Rechaza H_0. $\alpha = 0.05$.

Zona de rechazo: Para todo valor de probabilidad mayor o igual a 0.05, se acepta H_0 y se rechaza H_i.

Cálculo de grados de libertad: $gl = N - 1 = 24$

La efectividad del algoritmo genético mostrada en la tabla 6, se mide dependiendo el porcentaje de acierto respecto a los requerimientos nutricionales necesarios mencionados en la sección 3.2.3.

N	Efectividad del Prototipo %	Valor esperado %	*X*	$\overline{X}$	***X***
1	98.17	90	8.17	2.3872	5.69872384
2	97.61	90	7.61	1.8272	3.33865984
3	97.48	90	7.48	1.6972	2.88048784
4	97.93	90	7.93	2.1472	4.61046784
5	98.32	90	8.32	2.5372	6.43738384
6	97.91	90	7.91	2.1272	4.52497984
7	96.7	90	6.7	0.9172	0.84125584
8	96.4	90	6.4	0.6172	0.38093584
9	94.4	90	4.4	-1.3828	1.91213584
10	90.33	90	0.33	-5.4528	29.73302784
11	94.09	90	4.09	-1.6928	2.86557184

12	96.74	90	6.74	0.9572	0.91623184
13	92.6	90	2.6	-3.1828	10.13021584
14	94.17	90	4.17	-1.6128	2.60112384
15	93.47	90	3.47	-2.3128	5.34904384
16	94.07	90	4.07	-1.7128	2.93368384
17	93.29	90	3.29	-2.4928	6.21405184
18	94.59	90	4.59	-1.1928	1.42277184
19	95.35	90	5.35	-0.4328	0.18731584
20	96.23	90	6.23	0.4472	0.19998784
21	94.61	90	4.61	-1.1728	1.37545984
22	96.79	90	6.79	1.0072	1.01445184
23	98.12	90	8.12	2.3372	5.46250384
24	98.34	90	8.34	2.5572	6.53927184
25	96.86	90	6.86	1.0772	1.16035984
				$\sum(X-\bar{X})^2 =$	108.730104

Tabla 6. Pruebas del algoritmo genético.

Cálculo de la media aritmética.

$$\overline{X} = \frac{\sum X}{N} = \frac{144.57}{25} = 5.7828$$

Cálculo de la desviación estándar.

$$S = \sqrt{\frac{\sum(X-\bar{X})^2}{N-1}} = \sqrt{\frac{108.7301}{24}} = 2.1284$$

Cálculo del valor estadístico t:

$$t = \frac{\bar{X}}{\frac{S}{\sqrt{N}}} = \frac{5.7828}{\frac{2.1284}{4.8989}} = 13.5843$$

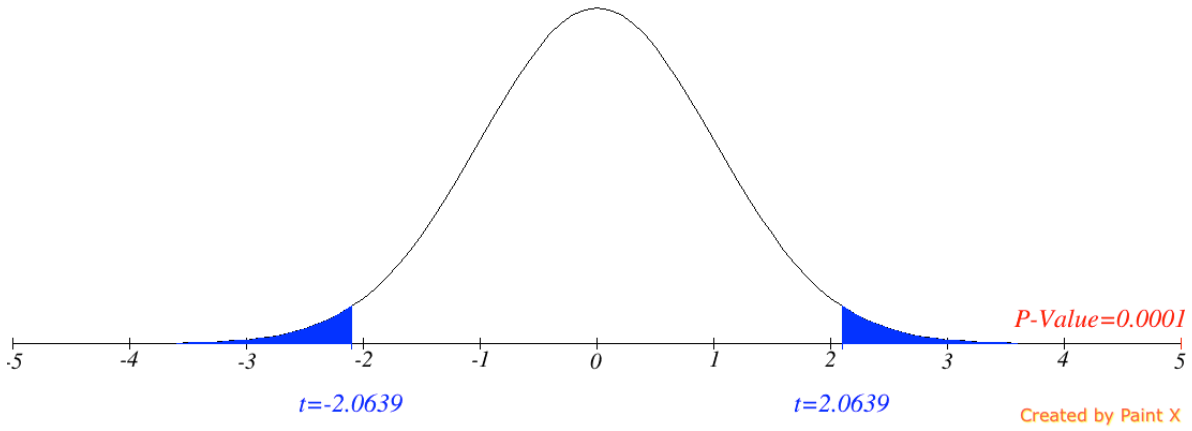


Figura 18. Regiones de aceptación y rechazo en el contraste de hipótesis.

El valor obtenido de t fue comparado con los valores críticos de la distribución t , se observa que a una probabilidad de 0.05 corresponde 2.262 de t. Por lo tanto , el cálculo tiene una probabilidad menor a 0.05.

Decisión: Como t es de 13.5843, con 24 grados de libertad, tiene un valor de probabilidad menor que 0.05 se acepta H_i y se rechaza H_0.

$H_i =$ El algoritmo genético propuesto es capaz de proporcionar un plan alimenticio balanceado para un paciente con DM2.

4.2.2 Análisis de la variable independiente.

Para el análisis de la variable independiente se utilizó una función exponencial que nos permite determinar el nivel de confiabilidad del Modelo considerando la siguiente escala.

80%-100%	Si es muy satisfactorio.
51%-79%	Si es satisfactorio.
10%-50%	No cumple los requisitos.

Tabla 7. Escala de efectividad del modelo de algoritmo genético.

A continuación se muestra la evaluación mediante la función exponencial:

$$f(t) = e^{-\lambda t}, con\ t \geq 0$$

Donde:

$\lambda = Representa\ el\ número\ de\ recomendaciones\ alimenticias\ acertadas.$

$t = tiempo\ que\ trabaja\ el\ Modelo\ de\ Algoritmo\ Genético.$

Si se inicia en el instante t_0:

$$P[T \leq t] = F(t)$$

Por lo tanto la probabilidad de trabajo sin fallas del modelo en el tiempo t es:

$$R(t) = P[T > t] = 1 - F(t) = 1 - e^{-\lambda t}$$

De los resultados obtenidos por el Modelo de Algoritmo Genético tenemos que de 25 recomendaciones alimenticias generadas el modelo falló en 1 debido a que los datos

de entrada no eran balanceados, esto quiere decir que los alimentos proporcionados al algoritmo genético no contenían todos los grupos de la pirámide alimenticia, se tomo como ejemplo que 25 recomendaciones correspondían a un tiempo de trabajo de 30 días.

La confiabilidad del Modelo de Algoritmo Genético para generar un plan alimenticio adecuado para un paciente con DM2 será:

$$R(t) = 1 - e^{-\left(\frac{24}{25}\right)30} = 1 - e^{-(0.92)30} = 0.99$$

Lo cual indica que modelo de algoritmo genético en 99% confiable.

4.2.3 Prueba del Control difuso.

El control difuso perteneciente al Agente Evaluación de la sección 3.4, es el encargado de juzgar la efectividad de la recomendación alimenticia. En particular se tomo como ejemplo el siguiente escenario:

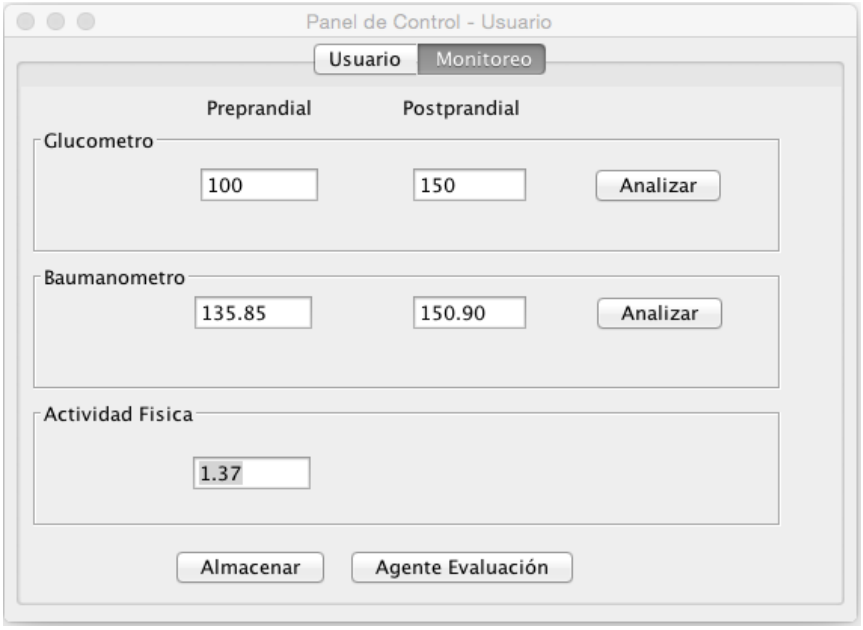


Figura 19. Escenario de prueba.

Obteniendo los Factores de Regulación tenemos:

- Factor de Regulación Glucosa: - 50
- Factor de Regulación de Presión arterial: -15.05

4.2.3.1 Fuzzyficación (Singleton)

Para el factor de Regulación de glucosa tenemos:

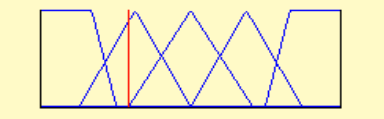


Figura 20. Valor singleton del Factor de regulación de glucosa.

El valor Singleton del "Factor de regulación de glucosa = -50" (ver figura 20) se corresponde un grado de verdad de 0.9 para el valor difuso "Factor de regulación de glucosa Buena (B)".

Para el factor de Regulación de presión arterial tenemos:

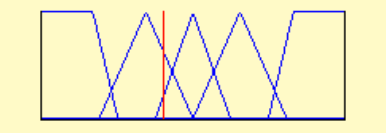


Figura 21. Valor singleton del Factor de regulación de presión arterial.

El valor Singleton del "Factor de regulación de presión arterial = -15.05" (ver figura 21) se corresponde un grado de verdad de 0.602 para el valor difuso "Factor de regulación de presión arterial Buena (B)" y un grado de verdad de 0.24 para el valor difuso "Factor de regulación de presión arterial Normal (N)".

4.2.3.2 Reglas Activadas

El escenario propuesto activa las reglas 6 y 7 mostradas en la figura 22, del conjunto de reglas propuestas en la sección (Agente Evaluación):

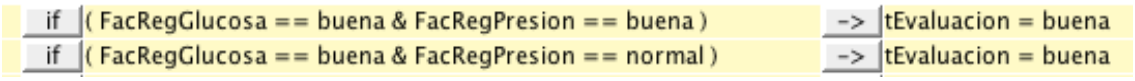
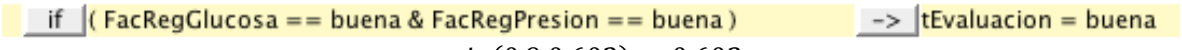


Figura 22. Reglas Activas

4.2.3.4 Modus ponens Difuso



$$min(0.9,0.602) \rightarrow 0.602$$

if (FacRegGlucosa == buena & FacRegPresion == normal) -> tEvaluacion = buena

$$min(0.9,0.24) \rightarrow 0.24$$

4.3.2.5 Agregación y defuzzyficación

Variable de Control Evaluación: Buena 66.66 (ver figura 23).

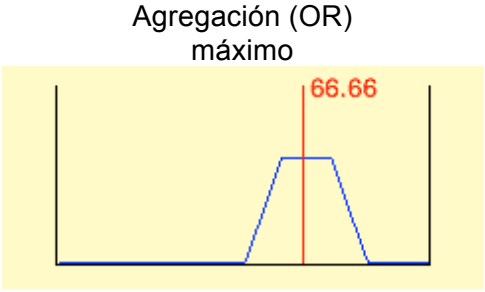


Figura 23. Valor de la variable de control evaluación.

La figura 24 muestra las curvas de conmutación del sistema de inferencia difuso, en las cuales aparecen las entradas del factor de regulación de glucosa (FacRegGlucosa) y del factor de regulación de presión arterial (FagRegPresion) y la salida que es la evaluación de la recomendación (tEvaluacion) por medio de los factores anteriores.

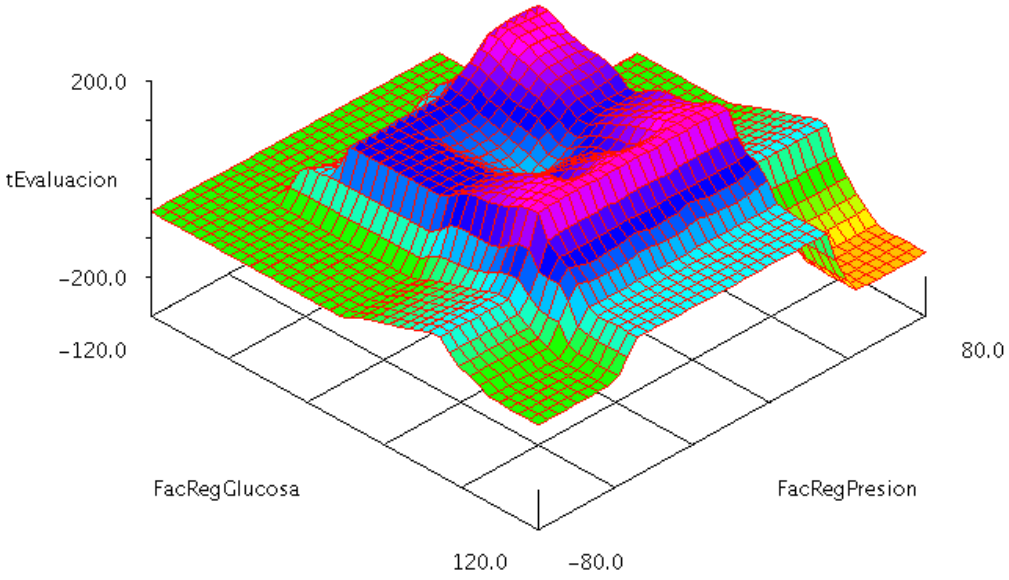


Figura 24. Curvas de conmutación del sistema de inferencia difuso.

Capítulo V

5.1 CONCLUSIONES

Podemos concluir que es posible implementar un sistema multiagente, capaz de proporcionar una recomendación alimenticia adecuada para un paciente con DM2, un sistema multiagente tiene como ventaja la modularización de los procesos que integran este sistema. Como el sistema posteriormente será implantado en un dispositivo móvil y el poder de procesamiento en estos hoy en día es limitado, lograr dividir la solución, en este caso en tres partes, otorga una gran ventaja para el desarrollador como para el usuario ya que solo el Agente Interfaz debe ser adaptado a estos requerimientos sin afectar el funcionamiento de las otras partes del sistema multiagente.

También fue posible proporcionar al paciente con DM2 una herramienta útil para el almacenamiento de su automonitoreo mediante el Panel de Control Usuario, esta es una parte importante en el control de la DM2, debido a la gran cantidad de datos que el paciente con DM2 maneja diariamente respecto a sus mediciones de glucosa y presión arterial y actividad física.

Respecto a las preferencias alimenticias el sistema es capaz de proporcionar al usuario una lista de 8 618 alimentos divididos en 27 categorías, descritos por su índice nutrimental, o si lo desea, el usuario puede agregar algún alimento de su preferencia para ser almacenado. Estos alimentos pertenecen a la USDA National Nutrient Database ver. 27 [16].

El algoritmo genético es capaz de proporcionar una recomendación alimenticia tomando en cuenta las preferencias alimenticias del usuario. Con un 94 – 98 % de efectividad. Otro aspecto importante del algoritmo es su tiempo de ejecución, que no sobrepasa los 60s.

Fue posible juzgar la calidad de la recomendación alimenticia proporcionada, mediante un sistema difuso, que toma en cuenta el factor de regulación de glucosa y presión arterial generado comparando los automonitoreos prepandial y postprandial.

Se proporcionó al usuario una serie de alertas, encargadas de avisar al paciente en el caso que alguna de las lecturas de su automonitoreo de glucosa y presión arterial llegue a un rango peligroso para su salud y desemboque en una complicación de su padecimiento en particular la DM2, en el caso de la glucosa hiperglucemia o hipoglucemia y en la presión arterial sufrir hipertensión.

5.2 TRABAJO A FUTURO

Para mayor comodidad del usuario, se plantea migrar el Agente Interfaz a un dispositivo móvil. Esto implicará que el usuario tenga disponible la recomendación alimenticia, la evaluación de su recomendación y la gestión de su automonitoreo en cualquier lugar donde se encuentre.

En particular, los datos de automonitoreo de glucosa , presión arterial y actividad física deben ser ingresados manualmente por el usuario. Se plantea mejorar esta parte agregando el uso de sensores para cada uno de estos aspectos, existen sensores portátiles seguros y confiables que son capaces de interactuar con un dispositivo móvil, donde automáticamente el Agente Interfaz llevaría a cabo la gestión de estos datos.

En el momento de agregar las preferencias alimenticias se debe contar con un mínimo de conocimiento para saber que un plan alimenticio debe contener todos los grupos de la pirámide alimentaria: el mínimo de azucares posible, lácteos, aceites y grasas, verduras, frutas, pescados, carnes huevos y leguminosas secas y cereales, papas y leguminosas frescas. El balance de los datos para generar el plan alimenticio, si el usuario no cuenta con este conocimiento, puede ser solucionado si el Agente Dieta es capaz de proporcionar todos estos grupos de alimentos si el usuario se olvida de agregar alguno, mediante una dieta base que los contenga.

Proporcionar otro tipo de unidades de medida ayudaría al usuario a facilitar la selección de sus preferencias alimenticias. En particular se utilizó como medida estándar para describir el índice nutricional de cada alimento, los gramos para alimentos solidos y mililitros para los líquidos. Se plantea aumentar estas unidades a otras también comúnmente utilizadas como por ejemplo para los solidos: tazas y porciones, y para líquidos: teaspoon, tablespoon, copas y tazas.

Proporcionar diferentes menús alimenticios, evitaría el hastío alimentario para un paciente que necesita llevar un régimen alimenticio especialmente estricto como en este caso un paciente con DM2, es de gran importancia debido a que la mayoría desiste de comer sanamente, por el cansancio que implica comer lo mismo todos los días.

Se plantea que para el caso del Agente Dieta y el Agente Evaluación estén alojados en la nube donde el usuario por medio de su dispositivo móvil pueda acceder a ellos desde cualquier parte donde se encuentre siempre y cuando exista una conexión de datos. La razón para esto es, que en el caso de estos Agentes los procesos que llevan a cabo necesitan un poder computacional que en este momento un dispositivo móvil común no puede proporcionar.

5.3 REFERENCIAS BIBLIOGRÁFICAS

[1]. Fundación Mexicana para la Salud (Funsalud), 2006, «*La salud en México 2006-2012: visión de Funsalud*», México, Funsalud.

[2]. Cordova-Villalobos, José Ángel et al. «*Las enfermedades crónicas no transmisibles en México: sinopsis epidemiológica y prevención integral. Salud pública Méx* [online]». 2008, vol.50, n.5, pp. 419-427. ISSN 0036-3634.

[3]. «*Estadisticas a propósito del día mundial de la diabetes*» *INEGI Aguascalientes*, 2013.

[4]. «*Global views of potential on móvil health solutions to address key healthcare challenges in chronic disease [Internet]*». Connected Living. [Cited 2013 Jun 25].

[5]. Secretaría de prevención y promoción de la salud. «*DGE Boletín Epidemiológico Diabetes Mellitus tipo 2, primer trimestre del 2013.*».

[6]. Sahar, Farrukh, «*Web Based Intelligent Decision Support System for Type 2 Diabetes Patients (September 6, 2013)* ». IACSIT International Journal of Engineering and Technology, Vol. 5, No. 5, October 2013, Forthcoming. Available at SSRN: http://ssrn.com/abstract=2321520.

[7]. C. Lee, M. Wang, H. Li, W. Chen, «Intelligent Ontological Agent for Diabetic Food Recommendation.» *IEEE International Conference of Fuzzy Systems*, 2008.

[8]. Ö. Kafali, S.Bromuri, M. Sindlar, T. van der Weide, E. Aguilar, U. Schaechtle, B. Alves, D. Zufferey, E. Rodriguez, M. Schumancher, K. Sthatis. «COMMODITY12: A smarth e-Health Enviroment for Diabetes Magnament.» *Journal of Ambient Intelligent and Smart Enviroments*, 2013.

[9]. D. Campos-Delgado, M. Hernández-Ordoñez, R. Fermat, A. Gordillo-Moscoso., «Fuzzy-Based Controller for Glucose Regulation in Type-1 Diabetic Patients by Subcutaneous Route.» *IEEE Transactions on Biomedical Engenieering, 52(11)*, 2006.

[10]. Quinn CC. «*WellDoc mobile diabetes management randomized controlled trial: change in clinical and behavioral outcomes and patient and physician satisfaction*». Diabetes Technol Ther. 2008 Jun ;10(3):160-8. doi: 10.1089/dia.2008.0283.

[11]. R. Schuman, S. Bromuri, J. Krampf, M. Schumacher., «Agent Based Monitoring of Gestational Diabetes Mellitus.» *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012), Valencia, Spain. (www.ifaamas.org)*, 2012.

[12]. K. Kurran, E. Nichols, R. Fermat, E. Xie, R. Harper, «*An intensive Insulinotherapy Mobile Phone Application Built on Artificial Intelligence Techniques.*» Journal of diabetes Science and Tecnology*, 4(1)*, 2010.

[13]. S. Bin-Sabbhar, A. Al-Roddhan, «*An Integrated Monitoring System for Mannaging Diabetes Patients Using Mobile Computing Technology.*» Proceddings of the World Congress on Engenieering and Computer Science*, Vol 1*, 2012.

[14]. E. Årsand, N. Tatara, G. Østengen, . «*Mobile Phone-Based Self-Magnament Tools for type 2 Diabetes: The few touch application.*» Journal of diabetes Science and Tecnology*, 4(2)*, 2010.

[15]. S. van der Weegen, R. Verwey, M. Spreeuwenberg, H. Tange, T. van der Weijden, L. de Witte, «*The Development of a Mobile Monitoring and Feedback Tool to Stimulate Physical Activity of People With a Chronic Disease in Primary Care: A User-Centered Design.*» JMIR MHEALTH AND UHEALTH , 2012.

[16]. R. Lopez, A. Chagpar, R. White, M. Mclean, M. Trudel, J. Cafazzo, A. Logan, «*Usability of a Diabetes*

Telemagnament System.» Journal of Clinical Engeneering*, 34(3),* 2009.

[17]. D. Pandey, N. Kumar, V. Pandey, «*SMS-Based System for Type-II Diabetes (NIDDM) Management.*» Health Informatics – An International Journal*, 1(1)*, 2012.

[18]. C. H. Yu, «*Designing and evaluating a web-based self- management site for patients with type 2 diabetes - systematic website development and study protocol.*» BMC Medical Informatics and Decision Making 2012, 12:57.

[19]. O. Salameh, «*A Mobile phone SMS-based system for diabetes self management.*» International Arab Journal of e-Tecnology, Vol. 2, No. 3, January 2012.

[20]. K. Blanchet, «*Telehealth and Diabetes Monitoring.*» Mary Ann Liebert, Inc. • Vol. 14 No. 8 • October 2008 Telemedicine and e-Health.

[21]. F.Sahar, «*Web based intelligent decision support system for type 2 diabetes patient*.» IACSIT International Journal of Ingenieering and Technology, 5(5), 2013.

[22]. Health Pia, Inc, «*The HealthPia GlucoPackTM Diabetes Phone: A Usability Study*.» Diabetes Technology & Therapeutics Volume 9, Number 2, 2007.

[23]. O. Ferrer-Roca, «*Mobile phone text messaging in the management of diabetes*.» Journal of Telemedicine and Telecare 2004; 10: 282–286.

[24]. V. Franklin, «"*Sweet Talk": Text Messaging Support for Intensive Insulin Therapy for Young People with Diabetes*». Diabetes Technology & Therapeutics Volume 5, Number 6, 2003.

[25]. D. Hanauer, «*Computarized automated remainder diabetes system (Cards): Using web and wireless pone technology to improve diabetes compliance*.». Massachusetts Institute of Technology, June, 2004.

[26]. B. Rami, «*Telemedical support to improve glycemic control in adolescents with type 1 diabetes mellitus*.». Eur J Pediatr (2006) 165: 701–705 DOI 10.1007/s00431-006-0156-6.

[27]. M. Rossi, «*Diabetes Interactive Diary: A New Telemedicine System Enabling Flexible Diet and Insulin Therapy While Improving Quality of Life*». Diabetes Care, Vol. 33, No. 1, January 2010.

[28]. M. Bell, «*Mobile Phone-Based Video Messages for Diabetes Self-Care Support*». Journal of Diabetes Science and Technology, Vol. 6, Issue 2, March 2012.

[29]. G. Mckay, «*Internet-Based Diabetes Self-Management and Support: Initial Outcomes From the Diabetes Network Project*». Rehabilitation Psychology, Vol. 47, No. 1, 31–48.

[30]. D. Gammon, «*Parent-Child Interaction Using a Mobile and Wireless System for Blood Glucose Monitoring*». J Med Internet Res 2005; 7 (5): e57.

[31]. A. Logan, «Mobile Phone–Based Remote Patient Monitoring System for Management of Hypertension in Diabetic Patients». AJH 2007; 20:942–948.

[32]. USDA United States Departament of Agriculture, «National nutrient database SR27.» *http://www.ars.usda.gov/Services/docs.htm?docid=24912.*

[33]. D. Miffin, «*A new predictive equation for resting energy expenditure in healthy individuals*». Am J Clin Nutr. vol. 51 No. 2 (February 1990) 241-247.

[34]. C. Zehnder, «*SODIO, POTASIO E HIPERTENSIÓN ARTERIAL*». REV. MED. CLIN. CONDES - 2010; 21(4) 508-515.

[35]. O. Lagunes, L. Morales, «*Monitor y Recomendador de plan alimenticio móvil, para regular pacientes con diabetes tipo 2*.» ITSM, 2013.

[36]. M. Zárate, «*Manual de procedimientos estandarizados para la vigilancia epidemiològica de la Diabetes Mellitus Tipo 2*.» Secretaría de Salud, SNVE, Septiembre 2012.

[37]. Medline Plus, «*Diabetes Tipo 2*.» Medline Plus, Agosto 2014.

[38]. J. Pozzi, «*Consejos sobre actividad física en la diabetes tipo 2*.» Servicio Diabetología, Hospital Córdoba, Agosto 2014. http://diabeteshospitalcordoba.com/pacientes/consejos-sobre-actividad-fisica/.

[39]. R. G. Schafer, B. Bohannon, M. Franz, J. Freeman, A. Holmes, S. McLaughlin, L. B. Haas, D. F. Kruger, R. A. Lorenz, and M. M. McMahon, «*Translation of the diabetes nutrition recommendations for health care institutions.*» *Diabetes Care*, vol. 20, pp. 96–105, 1997.

[40]. B.K. Ambati, J. Ambati, M.M. Mokhtar (1991). «*Heuristic combinatorial optimization by simulated Darwinian evolution: A polynomial time algorithm for the traveling salesman problem*», Biological Cybernetics, 65, 31-35.

[41] A. Will, «*Algortimos genéticos y optimización heurística*», Grupo de Aplicaciones de Inteligencia artificial, Universidad Nacional de Tucuman.

[42] C. Sánchez, «*Caracterización del estado nutricional de los niños y niñas menores de cinco años, beneficiarios del programa desayunos infantiles con amor del municipio de Mosquera*», Pontificia Universidad Javerian, Facultad de ciencias Departamento de Nutrición y Bioquímica, Carrera de Nutrición y Dietética, Bogotá, 2012.

[43] American Diabetes Association, «*Nutrition recommendation and principles for people with diabetes mellitus*.», Diabetes Care, 1997;20 (Supl 1):S14-S17.

ANEXOS

A.1 Programas.

A.1.1 Creación y comunicación de los Agentes.

El siguiente código pertenece a las clases Jade.java, JadeCal.java y JadeEval.java, es la encargadas de crear los respectivos agentes Interfaz, Dieta y Evaluación y proporcionarles los métodos para la comunicación entre ellos.

```
1  package jade;
2
3  import jade.gui.GuiAgent;
4  import jade.gui.GuiEvent;
5  import jade.core.AID;
6
7  import jade.core.behaviours.CyclicBehaviour;
8  import jade.lang.acl.ACLMessage;
9  import jade.lang.acl.MessageTemplate;
10
11 import java.awt.*;
12 /**
13  *
14  * @author omarlagunes
15  */
16 public class Jade extends GuiAgent {
17    VentanaChat vChat = new VentanaChat(this);
18    login vLogin = new login(this);
19    public void setup() {
20       CyclicBehaviour cb = new TalkBehaviour(this);
21       addBehaviour(cb);
22       //if (packFrame){
23       //    vChat.pack();
24       // }
25       // else{
26       //     vChat.validate();
27       // }
28       Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
29       Dimension frameSize = vChat.getSize();
30       vChat.setLocation((screenSize.width - frameSize.width)/2,(screenSize.height - frameSize.height)/2 );
31       vChat.setVisible(true);
32       vLogin.setVisible(true);
33    }
34
35    public void onGuiEvent(GuiEvent ge) {
```

```
36        String receiverName = ge.getParameter(0).toString();
37        String msgContent = ge.getParameter(1).toString();
38  //      System.out.print(ge.getParameter(0).toString()+ge.getParameter(1).toString());
39        ACLMessage toSend = new ACLMessage(ACLMessage.INFORM);
40        toSend.setContent(msgContent);
41        toSend.setPerformative(ACLMessage.INFORM);
42        toSend.addReceiver(new AID(receiverName, AID.ISLOCALNAME));
43        send(toSend);
44     }
45
46     public class TalkBehaviour extends CyclicBehaviour{
47        public TalkBehaviour (GuiAgent ga){
48           super(ga);
49        }
50           public void action() {
51              ACLMessage reply = receive(MessageTemplate.MatchPerformative(ACLMessage.INFORM));
52              if (reply!=null){
53                 String content = reply.getContent();
54                 String sender = reply.getSender().getName();
55                 vChat.taReceive.append("\n"+sender+": "+content);
56              }
57              else{
58                 block();
59              }
60           }
61
62     }
63  }
```

A.1.2 Agente Interfaz

A.1.2.1 Acceso.

El siguiente código pertenece a la clase login.java, que es la ventana de acceso y su modulo para agregar usuarios, este lleva a cabo el control de acceso y creación de usuarios.

```
1  package jade;
2
3  import claseConectar.conectar;
4  import jade.gui.GuiAgent;
5  import java.sql.*;
6  import java.sql.Statement;
```

```
7  import java.util.logging.Level;
8  import java.util.logging.Logger;
9  import javax.swing.JOptionPane;
10
11 /**
12  *
13  * @author omarlagunes
14  */
15 public class login extends javax.swing.JFrame {
16    //private GuiAgent owner;
17    /**
18     * Creates new form login
19     */
20    private GuiAgent owner;
21    public login(GuiAgent a) {
22       initComponents();
23       owner = a;
24    }
25
26    public void acceder(String usuario, String pass) throws Exception{
27       String sql = "SELECT * FROM Usuario WHERE nickName='"+usuario+"' && password='"+pass+"'";
28       Statement st;
29       String nick = "";
30       String passw = "";
31       try {
32          st = cn.createStatement();
33          ResultSet rs = st.executeQuery(sql);
34          while(rs.next()){
35             nick=rs.getString("nickName");
36             passw=rs.getString("password");
37             System.out.print(rs);
38             if(nick.equals(usuario)&&passw.equals(pass)){
39                this.setVisible(false);
40                JOptionPane.showMessageDialog(null, "Bienvenido");
41                VentanaChat vChat= new VentanaChat(owner);
42                cpUsuario vcpUsuario = new cpUsuario(nick, owner);
43 //                cpDiet vDiet = new cpDiet(owner);
44                vcpUsuario.setVisible(true);
45 //                vChat.setVisible(true);
46 //                vDiet.setVisible(true);
47             }
```

```
48              else{
49                  JOptionPane.showMessageDialog(null, "Usuario o contraseña incorrectos");
50              }
51          }
52      } catch (SQLException ex) {
53          Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
54      }
55  }
56
57
58  /**
59   * This method is called from within the constructor to initialize the form.
60   * WARNING: Do NOT modify this code. The content of this method is always
61   * regenerated by the Form Editor.
62   */
63  @SuppressWarnings("unchecked")
64  // <editor-fold defaultstate="collapsed" desc="Generated Code">
65  private void initComponents() {
66
67      jLabel1 = new javax.swing.JLabel();
68      jLabel2 = new javax.swing.JLabel();
69      pass = new javax.swing.JPasswordField();
70      user = new javax.swing.JTextField();
71      login = new javax.swing.JButton();
72      salir = new javax.swing.JButton();
73      crear = new javax.swing.JButton();
74
75      setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
76      setTitle("Acceso");
77
78      jLabel1.setText("Usuario:");
79
80      jLabel2.setText("Contraseña:");
81
82      pass.setText("jPasswordField1");
83
84      login.setText("Login");
85      login.addActionListener(new java.awt.event.ActionListener() {
86          public void actionPerformed(java.awt.event.ActionEvent evt) {
87              loginActionPerformed(evt);
88          }
```

```
89      });
90
91      salir.setText("Salir");
92
93      crear.setText("Crear");
94      crear.addActionListener(new java.awt.event.ActionListener() {
95          public void actionPerformed(java.awt.event.ActionEvent evt) {
96              crearActionPerformed(evt);
97          }
98      });
99
100     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
101     getContentPane().setLayout(layout);
102     layout.setHorizontalGroup(
103         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
104         .addGroup(layout.createSequentialGroup()
105             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
106                 .addGroup(layout.createSequentialGroup()
107                     .addGap(58, 58, 58)
108                     .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
109                         .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 54,
javax.swing.GroupLayout.PREFERRED_SIZE)
110                         .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 77,
javax.swing.GroupLayout.PREFERRED_SIZE))
111                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
112                     .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
113                         .addComponent(pass)
114                         .addComponent(user)))
115                 .addGroup(layout.createSequentialGroup()
116                     .addGap(39, 39, 39)
117                     .addComponent(login)
118                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
119                     .addComponent(salir)
120                     .addGap(12, 12, 12)
121                     .addComponent(crear)))
122             .addContainerGap(36, Short.MAX_VALUE))
123     );
124     layout.setVerticalGroup(
125         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
126         .addGroup(layout.createSequentialGroup()
127             .addGap(35, 35, 35)
128             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
129                 .addComponent(jLabel1)
```

```
                .addComponent(user, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(21, 21, 21)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel2)
                .addComponent(pass, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(18, 18, 18)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(login)
                .addComponent(salir)
                .addComponent(crear))
            .addContainerGap(44, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void loginActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String a = "";
        String b = "";
        a = user.getText();
        b = pass.getText();
        try {
            acceder(a,b);
        } catch (Exception ex) {
            Logger.getLogger(login.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void crearActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        crearUsr crear = new crearUsr();
        crear.setVisible(true);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
```

```
171         /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
172          * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
173          */
174         try {
175             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
176                 if ("Nimbus".equals(info.getName())) {
177                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
178                     break;
179                 }
180             }
181         } catch (ClassNotFoundException ex) {
182             java.util.logging.Logger.getLogger(login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
183         } catch (InstantiationException ex) {
184             java.util.logging.Logger.getLogger(login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
185         } catch (IllegalAccessException ex) {
186             java.util.logging.Logger.getLogger(login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
187         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
188             java.util.logging.Logger.getLogger(login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
189         }
190         //</editor-fold>
191 
192         /* Create and display the form */
193     }
194 
195 
196     // Variables declaration - do not modify
197     private javax.swing.JButton crear;
198     private javax.swing.JLabel jLabel1;
199     private javax.swing.JLabel jLabel2;
200     private javax.swing.JButton login;
201     private javax.swing.JPasswordField pass;
202     private javax.swing.JButton salir;
203     private javax.swing.JTextField user;
204     // End of variables declaration
205     String db = "jdbc:mysql://localhost/Agentes";
206     conectar cc = new conectar();
207     Connection cn = cc.conexion(db);
208 }
```

A.1.2.2 Panel de Control Usuario

El siguiente código pertenece a la clase cpUsuario.java, que es el panel de control usuario, en el se encuentran los módulos necesarios para crear el expediente personal del

paciente, obtener y almacenar sus automonitoreos y desde aquí se pueden solicitar las evaluaciones antropométrica y bioquímica del usuario.

```
1  package jade;
2
3  import claseConectar.conectar;
4  import jade.core.AID;
5  import jade.gui.GuiAgent;
6  import jade.gui.GuiEvent;
7  import jade.lang.acl.ACLMessage;
8  import java.sql.Connection;
9  import java.sql.ResultSet;
10 import java.sql.SQLException;
11 import java.sql.Statement;
12 import java.util.logging.Level;
13 import java.util.logging.Logger;
14 import javax.swing.JOptionPane;
15
16 /**
17  *
18  * @author omarlagunes
19  */
20 public class cpUsuario extends javax.swing.JFrame {
21     private String User;
22     String S= "";
23     String E= "";
24     private GuiAgent owner;
25     public final int SENT_TYPE = 0;
26     /**
27      * Creates new form cpUsuario
28      */
29     public cpUsuario(String Usuario, GuiAgent a) {
30         initComponents();
31         owner = a;
32         User = Usuario;
33         String sql = "SELECT * FROM Usuario WHERE nickName='"+User+"'";
34         Statement st;
35         String I= "";
36         try {
37             st = cn.createStatement();
38             ResultSet rs = st.executeQuery(sql);
39             rs.next();
```

```
40          I = Integer.toString(rs.getInt("idUsuario"));
41          S = rs.getString("Sexo");
42          E = Integer.toString(rs.getInt("Edad"));
43          lId.setText(I);
44          lNombre.setText(User);
45          lEdad.setText(E);
46          lSexo.setText(S);
47      } catch (SQLException ex) {
48          Logger.getLogger(cpUsuario.class.getName()).log(Level.SEVERE, null, ex);
49      }
50  }
51
52  /**
53   * This method is called from within the constructor to initialize the form.
54   * WARNING: Do NOT modify this code. The content of this method is always
55   * regenerated by the Form Editor.
56   */
57  @SuppressWarnings("unchecked")
58  // <editor-fold defaultstate="collapsed" desc="Generated Code">
59  private void initComponents() {
60
61      jTabbedPane1 = new javax.swing.JTabbedPane();
62      jPanel1 = new javax.swing.JPanel();
63      jPanel2 = new javax.swing.JPanel();
64      jLabel1 = new javax.swing.JLabel();
65      lId = new javax.swing.JLabel();
66      jLabel3 = new javax.swing.JLabel();
67      lNombre = new javax.swing.JLabel();
68      jLabel5 = new javax.swing.JLabel();
69      lEdad = new javax.swing.JLabel();
70      jLabel7 = new javax.swing.JLabel();
71      lSexo = new javax.swing.JLabel();
72      jPanel4 = new javax.swing.JPanel();
73      jLabel2 = new javax.swing.JLabel();
74      jLabel4 = new javax.swing.JLabel();
75      jLabel6 = new javax.swing.JLabel();
76      jLabel8 = new javax.swing.JLabel();
77      peso = new javax.swing.JTextField();
78      talla = new javax.swing.JTextField();
79      iHbAc1 = new javax.swing.JTextField();
80      iCad = new javax.swing.JTextField();
```

```
81      jButton1 = new javax.swing.JButton();
82      jPanel5 = new javax.swing.JPanel();
83      jScrollPane1 = new javax.swing.JScrollPane();
84      moAn = new javax.swing.JTextArea();
85      jPanel3 = new javax.swing.JPanel();
86      jPanel6 = new javax.swing.JPanel();
87      preGluc = new javax.swing.JTextField();
88      posGluc = new javax.swing.JTextField();
89      jButton4 = new javax.swing.JButton();
90      jPanel8 = new javax.swing.JPanel();
91      prePre = new javax.swing.JTextField();
92      prePost = new javax.swing.JTextField();
93      jButton5 = new javax.swing.JButton();
94      jPanel7 = new javax.swing.JPanel();
95      activ = new javax.swing.JTextField();
96      jLabel9 = new javax.swing.JLabel();
97      jLabel10 = new javax.swing.JLabel();
98      jButton2 = new javax.swing.JButton();
99      jButton3 = new javax.swing.JButton();
100
101     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
102     setTitle("Panel de Control - Usuario");
103
104     jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder("Registro"));
105
106     jLabel1.setText("Id:");
107
108     jLabel3.setText("Nombre:");
109
110     jLabel5.setText("Edad:");
111
112     jLabel7.setText("Sexo:");
113
114     javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
115     jPanel2.setLayout(jPanel2Layout);
116     jPanel2Layout.setHorizontalGroup(
117        jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
118        .addGroup(jPanel2Layout.createSequentialGroup()
119           .addContainerGap()
120           .addComponent(jLabel1)
121           .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
122             .addComponent(lId)
123             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
124             .addComponent(jLabel3)
125             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
126             .addComponent(lNombre)
127             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
128             .addComponent(jLabel5)
129             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
130             .addComponent(lEdad)
131             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
132             .addComponent(jLabel7)
133             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
134             .addComponent(lSexo)
135             .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
136         );
137         jPanel2Layout.setVerticalGroup(
138             jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
139             .addGroup(jPanel2Layout.createSequentialGroup()
140                 .addContainerGap()
141                 .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
142                     .addComponent(jLabel1)
143                     .addComponent(lId)
144                     .addComponent(jLabel3)
145                     .addComponent(lNombre)
146                     .addComponent(jLabel5)
147                     .addComponent(lEdad)
148                     .addComponent(jLabel7)
149                     .addComponent(lSexo))
150                 .addContainerGap(20, Short.MAX_VALUE))
151         );
152
153         jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder("Antropometría"));
154
155         jLabel2.setText("Peso (kg)");
156
157         jLabel4.setText("Estatura (mts)");
158
159         jLabel6.setText("HbAc1");
160
161         jLabel8.setText("IAF");
162
```

```
163         peso.addActionListener(new java.awt.event.ActionListener() {
164             public void actionPerformed(java.awt.event.ActionEvent evt) {
165                 pesoActionPerformed(evt);
166             }
167         });
168 
169         jButton1.setText("Evaluar");
170         jButton1.addActionListener(new java.awt.event.ActionListener() {
171             public void actionPerformed(java.awt.event.ActionEvent evt) {
172                 jButton1ActionPerformed(evt);
173             }
174         });
175 
176         javax.swing.GroupLayout jPanel4Layout = new javax.swing.GroupLayout(jPanel4);
177         jPanel4.setLayout(jPanel4Layout);
178         jPanel4Layout.setHorizontalGroup(
179             jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
180             .addGroup(jPanel4Layout.createSequentialGroup()
181                 .addContainerGap()
182                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
183                     .addGroup(jPanel4Layout.createSequentialGroup()
184                         .addGap(0, 0, Short.MAX_VALUE)
185                         .addComponent(jButton1)
186                         .addContainerGap())
187                     .addGroup(jPanel4Layout.createSequentialGroup()
188                         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
189                             .addComponent(jLabel2)
190                             .addComponent(peso, javax.swing.GroupLayout.PREFERRED_SIZE, 63,
javax.swing.GroupLayout.PREFERRED_SIZE))
191                         .addGap(58, 58, 58)
192                         .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
193                             .addGroup(jPanel4Layout.createSequentialGroup()
194                                 .addGap(14, 14, 14)
195                                 .addComponent(talla, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
196                                 .addGap(88, 88, 88)
197                                 .addComponent(iHbAc1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
198                             .addGroup(jPanel4Layout.createSequentialGroup()
199                                 .addComponent(jLabel4)
200                                 .addGap(89, 89, 89)
201                                 .addComponent(jLabel6)))
202                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 88,
Short.MAX_VALUE)
```

```
203                     .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
204                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel4Layout.createSequentialGroup()
205                             .addComponent(iCad, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
206                             .addGap(14, 14, 14))
207                         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel4Layout.createSequentialGroup()
208                             .addComponent(jLabel8)
209                             .addGap(38, 38, 38))))))
210         );
211
212         jPanel4Layout.linkSize(javax.swing.SwingConstants.HORIZONTAL, new java.awt.Component[] {iCad,
iHbAc1, peso, talla});
213
214         jPanel4Layout.setVerticalGroup(
215             jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
216             .addGroup(jPanel4Layout.createSequentialGroup()
217                 .addContainerGap()
218                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
219                     .addComponent(jLabel2)
220                     .addComponent(jLabel4)
221                     .addComponent(jLabel6)
222                     .addComponent(jLabel8))
223                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
224                 .addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
225                     .addComponent(peso, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
226                     .addComponent(talla, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
227                     .addComponent(iHbAc1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
228                     .addComponent(iCad, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
229                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 7, Short.MAX_VALUE)
230                 .addComponent(jButton1))
231         );
232
233         jPanel4Layout.linkSize(javax.swing.SwingConstants.VERTICAL, new java.awt.Component[] {iCad, iHbAc1,
peso, talla});
234
235         jPanel5.setBorder(javax.swing.BorderFactory.createTitledBorder("Evalucación Antropométrica"));
236
237         moAn.setColumns(20);
238         moAn.setRows(5);
239         jScrollPane1.setViewportView(moAn);
```

```
240
241         javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);
242         jPanel5.setLayout(jPanel5Layout);
243         jPanel5Layout.setHorizontalGroup(
244             jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
245             .addComponent(jScrollPane1)
246         );
247         jPanel5Layout.setVerticalGroup(
248             jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
249             .addGroup(jPanel5Layout.createSequentialGroup()
250                 .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 117,
javax.swing.GroupLayout.PREFERRED_SIZE)
251                 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
252         );
253
254         javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
255         jPanel1.setLayout(jPanel1Layout);
256         jPanel1Layout.setHorizontalGroup(
257             jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
258             .addGroup(jPanel1Layout.createSequentialGroup()
259                 .addContainerGap()
260                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
261                     .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
262                     .addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
263                     .addComponent(jPanel5, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
264                 .addContainerGap())
265         );
266         jPanel1Layout.setVerticalGroup(
267             jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
268             .addGroup(jPanel1Layout.createSequentialGroup()
269                 .addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
270                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
271                 .addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
272                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
273                 .addComponent(jPanel5, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
274         );
275
276         jTabbedPane1.addTab("Usuario", jPanel1);
277
```

```
278         jPanel6.setBorder(javax.swing.BorderFactory.createTitledBorder("Glucometro"));
279
280         jButton4.setText("Analizar");
281         jButton4.addActionListener(new java.awt.event.ActionListener() {
282             public void actionPerformed(java.awt.event.ActionEvent evt) {
283                 jButton4ActionPerformed(evt);
284             }
285         });
286
287         javax.swing.GroupLayout jPanel6Layout = new javax.swing.GroupLayout(jPanel6);
288         jPanel6.setLayout(jPanel6Layout);
289         jPanel6Layout.setHorizontalGroup(
290             jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
291             .addGroup(jPanel6Layout.createSequentialGroup()
292                 .addGap(104, 104, 104)
293                 .addComponent(preGluc, javax.swing.GroupLayout.PREFERRED_SIZE, 84,
javax.swing.GroupLayout.PREFERRED_SIZE)
294                 .addGap(57, 57, 57)
295                 .addComponent(posGluc, javax.swing.GroupLayout.PREFERRED_SIZE, 81,
javax.swing.GroupLayout.PREFERRED_SIZE)
296                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 37, Short.MAX_VALUE)
297                 .addComponent(jButton4)
298                 .addGap(30, 30, 30))
299         );
300         jPanel6Layout.setVerticalGroup(
301             jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
302             .addGroup(jPanel6Layout.createSequentialGroup()
303                 .addContainerGap()
304                 .addGroup(jPanel6Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
305                     .addComponent(preGluc, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
306                     .addComponent(posGluc, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
307                     .addComponent(jButton4))
308                 .addGap(0, 26, Short.MAX_VALUE))
309         );
310
311         jPanel6Layout.linkSize(javax.swing.SwingConstants.VERTICAL, new java.awt.Component[] {posGluc,
preGluc});
312
313         jPanel8.setBorder(javax.swing.BorderFactory.createTitledBorder("Baumanometro"));
314
315         jButton5.setText("Analizar");
316         jButton5.addActionListener(new java.awt.event.ActionListener() {
```

```
317             public void actionPerformed(java.awt.event.ActionEvent evt) {
318                 jButton5ActionPerformed(evt);
319             }
320         });
321 
322         javax.swing.GroupLayout jPanel8Layout = new javax.swing.GroupLayout(jPanel8);
323         jPanel8.setLayout(jPanel8Layout);
324         jPanel8Layout.setHorizontalGroup(
325             jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
326             .addGroup(jPanel8Layout.createSequentialGroup()
327                 .addGap(100, 100, 100)
328                 .addComponent(prePre, javax.swing.GroupLayout.PREFERRED_SIZE, 84,
javax.swing.GroupLayout.PREFERRED_SIZE)
329                 .addGap(61, 61, 61)
330                 .addComponent(prePost, javax.swing.GroupLayout.PREFERRED_SIZE, 81,
javax.swing.GroupLayout.PREFERRED_SIZE)
331                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
332                 .addComponent(jButton5)
333                 .addGap(29, 29, 29))
334         );
335         jPanel8Layout.setVerticalGroup(
336             jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
337             .addGroup(jPanel8Layout.createSequentialGroup()
338                 .addGroup(jPanel8Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
339                     .addComponent(prePost, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
340                     .addComponent(prePre, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
341                     .addComponent(jButton5))
342                 .addGap(0, 32, Short.MAX_VALUE))
343         );
344 
345         jPanel7.setBorder(javax.swing.BorderFactory.createTitledBorder("Actividad Fisica"));
346 
347         javax.swing.GroupLayout jPanel7Layout = new javax.swing.GroupLayout(jPanel7);
348         jPanel7.setLayout(jPanel7Layout);
349         jPanel7Layout.setHorizontalGroup(
350             jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
351             .addGroup(jPanel7Layout.createSequentialGroup()
352                 .addGap(99, 99, 99)
353                 .addComponent(activ, javax.swing.GroupLayout.PREFERRED_SIZE, 84,
javax.swing.GroupLayout.PREFERRED_SIZE)
354                 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
355         );
```

```
356         jPanel7Layout.setVerticalGroup(
357             jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
358             .addGroup(jPanel7Layout.createSequentialGroup()
359                 .addGap(16, 16, 16)
360                 .addComponent(activ, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
361                 .addContainerGap(18, Short.MAX_VALUE))
362         );
363
364         jLabel9.setText("Prepandreal");
365
366         jLabel10.setText("Postpandreal");
367
368         jButton2.setText("Almacenar");
369
370         jButton3.setText("Agente Evaluación");
371         jButton3.addActionListener(new java.awt.event.ActionListener() {
372             public void actionPerformed(java.awt.event.ActionEvent evt) {
373                 jButton3ActionPerformed(evt);
374             }
375         });
376
377         javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
378         jPanel3.setLayout(jPanel3Layout);
379         jPanel3Layout.setHorizontalGroup(
380             jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
381             .addGroup(jPanel3Layout.createSequentialGroup()
382                 .addGap(123, 123, 123)
383                 .addComponent(jLabel9)
384                 .addGap(60, 60, 60)
385                 .addComponent(jLabel10, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
386                 .addGap(207, 207, 207))
387             .addGroup(jPanel3Layout.createSequentialGroup()
388                 .addContainerGap()
389                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
390                     .addComponent(jPanel6, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
391                     .addComponent(jPanel8, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
392                     .addComponent(jPanel7, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
393                 .addGap(0, 0, Short.MAX_VALUE))
394             .addGroup(jPanel3Layout.createSequentialGroup()
```

```
395             .addGap(97, 97, 97)
396             .addComponent(jButton2)
397             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
398             .addComponent(jButton3)
399             .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
400         );
401         jPanel3Layout.setVerticalGroup(
402             jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
403             .addGroup(jPanel3Layout.createSequentialGroup()
404                 .addContainerGap()
405                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
406                     .addComponent(jLabel9)
407                     .addComponent(jLabel10))
408                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
409                 .addComponent(jPanel6, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
410                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
411                 .addComponent(jPanel8, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
412                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
413                 .addComponent(jPanel7, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
414                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
415                 .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
416                     .addComponent(jButton2)
417                     .addComponent(jButton3))
418                 .addContainerGap(8, Short.MAX_VALUE))
419         );
420
421         jTabbedPane1.addTab("Monitoreo", jPanel3);
422
423         javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
424         getContentPane().setLayout(layout);
425         layout.setHorizontalGroup(
426             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
427             .addComponent(jTabbedPane1)
428         );
429         layout.setVerticalGroup(
430             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
431             .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 393,
javax.swing.GroupLayout.PREFERRED_SIZE)
432         );
433
434         pack();
```

```
435     }// </editor-fold>
436
437     private void pesoActionPerformed(java.awt.event.ActionEvent evt) {
438         // TODO add your handling code here:
439     }
440
441     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
442         // TODO add your handling code here:
443         float [] vDat = new float[5] ;
444         double [] vDatAn = new double[7] ;
445         vDat[0] = Float.parseFloat(peso.getText());
446         vDat[1] = Float.parseFloat(talla.getText());
447         vDat[2] = Float.parseFloat(iHbAc1.getText());
448         vDat[3] = Float.parseFloat(iCad.getText());
449         vDat[4] = Float.parseFloat(E);
450         analisis anL = new analisis(vDat, S);
451         anL.calcular();
452         moAn.append("IMC: "+anL.calcular());
453         vDatAn = anL.vDatAn;
454         cpDiet vDiet = new cpDiet(owner, vDatAn);
455         vDiet.setVisible(true);
456     }
457
458     private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
459         // TODO add your handling code here:
460     }
461
462     private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
463         // TODO add your handling code here:
464         float preGl = 0.0f ,posGl = 0.0f;
465         preGl = Float.parseFloat(preGluc.getText());
466         if(preGl<70)
467             {
468                 JOptionPane.showMessageDialog ( null ,"Sus valores de Glucosa están en rango peligroso\n llamar al
    médico inmediatamente."
469                                 ,"Alerta",JOptionPane.ERROR_MESSAGE );
470             }
471     }
472
473     private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
474         // TODO add your handling code here:
475         float prePr = 0.0f ,posPr = 0.0f;
```

```
476         prePr = Float.parseFloat(prePre.getText());
477         if(prePr>176.110)
478           {
479               JOptionPane.showMessageDialog ( null ,"Sus valores de Presión Arterial están en rango peligroso\n
llamar al médico inmediatamente."
480                           ,"Alerta",JOptionPane.ERROR_MESSAGE );
481           }
482     }
483
484     /**
485      * @param args the command line arguments
486      */
487     public static void main(String args[]) {
488         /* Set the Nimbus look and feel */
489         //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
490         /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
491          * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
492          */
493         try {
494             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels())
{
495                 if ("Nimbus".equals(info.getName())) {
496                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
497                     break;
498                 }
499             }
500         } catch (ClassNotFoundException ex) {
501             java.util.logging.Logger.getLogger(cpUsuario.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
502         } catch (InstantiationException ex) {
503             java.util.logging.Logger.getLogger(cpUsuario.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
504         } catch (IllegalAccessException ex) {
505             java.util.logging.Logger.getLogger(cpUsuario.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
506         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
507             java.util.logging.Logger.getLogger(cpUsuario.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
508         }
509         //</editor-fold>
510
511         /* Create and display the form */
512     }
513
514     // Variables declaration - do not modify
```

```
515   private javax.swing.JTextField activ;
516   private javax.swing.JTextField iCad;
517   private javax.swing.JTextField iHbAc1;
518   private javax.swing.JButton jButton1;
519   private javax.swing.JButton jButton2;
520   private javax.swing.JButton jButton3;
521   private javax.swing.JButton jButton4;
522   private javax.swing.JButton jButton5;
523   private javax.swing.JLabel jLabel1;
524   private javax.swing.JLabel jLabel10;
525   private javax.swing.JLabel jLabel2;
526   private javax.swing.JLabel jLabel3;
527   private javax.swing.JLabel jLabel4;
528   private javax.swing.JLabel jLabel5;
529   private javax.swing.JLabel jLabel6;
530   private javax.swing.JLabel jLabel7;
531   private javax.swing.JLabel jLabel8;
532   private javax.swing.JLabel jLabel9;
533   private javax.swing.JPanel jPanel1;
534   private javax.swing.JPanel jPanel2;
535   private javax.swing.JPanel jPanel3;
536   private javax.swing.JPanel jPanel4;
537   private javax.swing.JPanel jPanel5;
538   private javax.swing.JPanel jPanel6;
539   private javax.swing.JPanel jPanel7;
540   private javax.swing.JPanel jPanel8;
541   private javax.swing.JScrollPane jScrollPane1;
542   private javax.swing.JTabbedPane jTabbedPane1;
543   private javax.swing.JLabel lEdad;
544   private javax.swing.JLabel lId;
545   private javax.swing.JLabel lNombre;
546   private javax.swing.JLabel lSexo;
547   public javax.swing.JTextArea moAn;
548   private javax.swing.JTextField peso;
549   private javax.swing.JTextField posGluc;
550   private javax.swing.JTextField preGluc;
551   private javax.swing.JTextField prePost;
552   private javax.swing.JTextField prePre;
553   private javax.swing.JTextField talla;
554   // End of variables declaration
555   String db = "jdbc:mysql://localhost/Agentes";
```

```
556     conectar cc = new conectar();
557     Connection cn = cc.conexion(db);
558 }
```

A.1.2.3 Panel de Control Dieta

El siguiente código pertenece a la clase cpDiet.java, que es el Panel de control dieta, desde aquí se seleccionan las preferencias alimenticias que serán consideradas en la recomendación, también desde aquí se solicita y muestra la recomendación alimenticia generada por el Agente Dieta.

```
1  package jade;
2
3  import claseConectar.conectar;
4  import jade.gui.GuiAgent;
5  import jade.gui.GuiEvent;
6  import java.sql.Connection;
7  import java.sql.ResultSet;
8  import java.sql.SQLException;
9  import java.sql.Statement;
10 import java.util.ArrayList;
11 import java.util.Vector;
12 import java.util.logging.Level;
13 import java.util.logging.Logger;
14 import javax.swing.tree.DefaultMutableTreeNode;
15 import javax.swing.tree.DefaultTreeModel;
16 import java.io.*;
17 import java.sql.DriverManager;
18 import java.text.DecimalFormat;
19 import javax.swing.JOptionPane;
20 import javax.swing.JProgressBar;
21 import javax.swing.table.DefaultTableModel;
22 /**
23  *
24  * @author omarlagunes
25  */
26 public class cpDiet extends javax.swing.JFrame {
27     /**
28      * Creates new form cpDiet
29      */
30     Vector <foodN> foods = new Vector();
31     DecimalFormat df = new DecimalFormat("#.##");
32     double [] vDatAna = new double[7];
33     private GuiAgent owner;
```

```
34    public final int SENT_TYPE = 0;
35    public Statement st;
36    DefaultTableModel model;
37    String info [] = new String [9];
38    double cant = 0.00d;
39    ResultSet rsP;
40    public cpDiet(GuiAgent a, double [] vDatAn){
41       initComponents();
42       owner = a;
43       vDatAna = vDatAn;
44 //       System.out.print(vDatAna[0]+" "+vDatAna[1]+" "+vDatAna[2]);
45       model = new DefaultTableModel();
46       jTable1.setModel(model);
47       model.addColumn("Id");
48       model.addColumn("Alimento");
49       model.addColumn("Carbohidratos");
50       model.addColumn("Proteinas");
51       model.addColumn("Grasas");
52       model.addColumn("Energía");
53       llenar();
54 //       cfo.start();
55 //       cn.close();
56    }
57
58    public final void llenar(){
59       String sql = "SELECT * FROM FD_GROUP";
60       try {
61          st = cn.createStatement();
62          ResultSet rs = st.executeQuery(sql);
63          int columnCount = (rs.getMetaData().getColumnCount())/2;
64          ArrayList list = new ArrayList();
65          while(rs.next())
66          {
67             Object row[] = {rs.getString(1), rs.getString(2)};
68             list.add(row);
69          }
70          Object jerarquia[] = list.toArray();
71          DefaultMutableTreeNode raiz  = proJerar(jerarquia);
72          DefaultTreeModel modArb = new DefaultTreeModel(raiz);
73          this.jTree1.setModel(modArb);
74          this.jTree1.setRootVisible(false);
```

```
75          } catch (SQLException ex) {
76              Logger.getLogger(cpDiet.class.getName()).log(Level.SEVERE, null, ex);
77          }
78      }
79
80      public DefaultMutableTreeNode proJerar(Object[] hierarchy) {
81          Statement st;
82          DefaultMutableTreeNode node = new DefaultMutableTreeNode(hierarchy[0]);
83          try {
84              int ctrow = 0;
85              int i = 0;
86              try {
87                  st = cn.createStatement();
88                  String sql = "SELECT FdGrp_Desc, FdGrp_CD from FD_GROUP";
89                  ResultSet rs = st.executeQuery(sql);
90
91                  while (rs.next()) {
92                      ctrow = rs.getRow();
93                  }
94                  String L1Nam[] = new String[ctrow];
95                  String L1Id[] = new String[ctrow];
96                  ResultSet rs1 = st.executeQuery(sql);
97                  while (rs1.next()) {
98                      L1Nam[i] = rs1.getString("FdGrp_Desc");
99                      L1Id[i] = rs1.getString("FdGrp_CD");
100                     i++;
101                 }
102                 DefaultMutableTreeNode child, grandchild;
103                 for (int childIndex = 0; childIndex < L1Nam.length; childIndex++) {
104                     child = new DefaultMutableTreeNode(L1Nam[childIndex]);
105                     node.add(child);//add each created child to root
106                     String sql2 = "SELECT Long_Desc from FOOD_DES where FdGrp_CD= '" + L1Id[childIndex] + "' ";
107                     ResultSet rs3 = st.executeQuery(sql2);
108                     while (rs3.next()) {
109                         grandchild = new DefaultMutableTreeNode(rs3.getString("Long_Desc"));
110                         child.add(grandchild);//add each grandchild to each child
111                     }
112                 }
113
114             } catch (Exception ex) {
115                 ex.printStackTrace();
```

```
116             }
117
118         } catch (Exception e) {
119         }
120
121         return (node);
122     }
123     /**
124      * This method is called from within the constructor to initialize the form.
125      * WARNING: Do NOT modify this code. The content of this method is always
126      * regenerated by the Form Editor.
127      */
128     @SuppressWarnings("unchecked")
129     // <editor-fold defaultstate="collapsed" desc="Generated Code">
130     private void initComponents() {
131
132         jPanel1 = new javax.swing.JPanel();
133         jLabel1 = new javax.swing.JLabel();
134         jLabel2 = new javax.swing.JLabel();
135         jLabel3 = new javax.swing.JLabel();
136         tProt = new javax.swing.JTextField();
137         tGras = new javax.swing.JTextField();
138         tCarbo = new javax.swing.JTextField();
139         jLabel4 = new javax.swing.JLabel();
140         tEner = new javax.swing.JTextField();
141         jScrollPane2 = new javax.swing.JScrollPane();
142         jTable1 = new javax.swing.JTable();
143         jTabbedPane1 = new javax.swing.JTabbedPane();
144         jScrollPane1 = new javax.swing.JScrollPane();
145         jTree1 = new javax.swing.JTree();
146         jPanel2 = new javax.swing.JPanel();
147         jScrollPane4 = new javax.swing.JScrollPane();
148         jTable2 = new javax.swing.JTable();
149         jButton3 = new javax.swing.JButton();
150         jButton1 = new javax.swing.JButton();
151         jLabel5 = new javax.swing.JLabel();
152         tCant = new javax.swing.JTextField();
153         bSend = new javax.swing.JButton();
154         jScrollPane3 = new javax.swing.JScrollPane();
155         jTextArea1 = new javax.swing.JTextArea();
156         jLabel6 = new javax.swing.JLabel();
```

```
157         jLabel7 = new javax.swing.JLabel();
158         jButton2 = new javax.swing.JButton();
159
160         setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
161         setTitle("Panel de Control - Dieta");
162
163         jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder("Información Nutrimental"));
164
165         jLabel1.setText("Proteina/gr");
166
167         jLabel2.setText("Grasa/gr");
168
169         jLabel3.setText("Carbohidratos/gr");
170
171         jLabel4.setText("Energia/gr");
172
173         javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
174         jPanel1.setLayout(jPanel1Layout);
175         jPanel1Layout.setHorizontalGroup(
176             jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
177             .addGroup(jPanel1Layout.createSequentialGroup()
178                 .addContainerGap(19, Short.MAX_VALUE)
179                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
180                     .addComponent(jLabel1)
181                     .addGroup(jPanel1Layout.createSequentialGroup()
182                         .addGap(6, 6, 6)
183                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
184                             .addComponent(jLabel2)
185                             .addComponent(tProt, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE)
186                             .addComponent(tGras, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE))))
187                 .addGap(22, 22, 22)
188                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
189                     .addComponent(jLabel3)
190                     .addComponent(jLabel4)
191                     .addGroup(jPanel1Layout.createSequentialGroup()
192                         .addGap(6, 6, 6)
193                         .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
194                             .addComponent(tEner, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE)
195                             .addComponent(tCarbo, javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE)))))
196         );
```

```
197         jPanel1Layout.setVerticalGroup(
198             jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
199             .addGroup(jPanel1Layout.createSequentialGroup()
200                 .addContainerGap()
201                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
202                     .addComponent(jLabel1)
203                     .addComponent(jLabel3))
204                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
205                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
206                     .addComponent(tProt, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
207                     .addComponent(tCarbo, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
208                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
209                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
210                     .addComponent(jLabel2)
211                     .addComponent(jLabel4))
212                 .addGap(18, 18, 18)
213                 .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
214                     .addComponent(tGras, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
215                     .addComponent(tEner, javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
216                 .addContainerGap())
217         );
218
219         jTable1.setModel(new javax.swing.table.DefaultTableModel(
220             new Object [][] {
221                 {null, null, null, null, null, null},
222                 {null, null, null, null, null, null},
223                 {null, null, null, null, null, null},
224                 {null, null, null, null, null, null}
225             },
226             new String [] {
227                 "Id", "Alimento", "Proteinas", "Carbohidratos", "Energía", "Grasas"
228             }
229         ));
230         jTable1.setToolTipText("");
231         jScrollPane2.setViewportView(jTable1);
232
233         jTree1.setBorder(javax.swing.BorderFactory.createTitledBorder("Alimentos"));
234         javax.swing.tree.DefaultMutableTreeNode treeNode1 = new
    javax.swing.tree.DefaultMutableTreeNode("root");
235         jTree1.setModel(new javax.swing.tree.DefaultTreeModel(treeNode1));
```

```
236         jTree1.addMouseListener(new java.awt.event.MouseAdapter() {
237             public void mouseClicked(java.awt.event.MouseEvent evt) {
238                 jTree1MouseClicked(evt);
239             }
240         });
241         jScrollPane1.setViewportView(jTree1);
242 
243         jTabbedPane1.addTab("Comidas- DB", jScrollPane1);
244 
245         jTable2.setModel(new javax.swing.table.DefaultTableModel(
246             new Object [][] {
247                 {null, null, null, null, null, null, null, null, null, null}
248             },
249             new String [] {
250                 "ID", "Grupo", "Descripcion", "Proteinas", "Grasas", "Carbohidratos", "Energia", "Glucosa", "Potasio", "Sodio"
251             }
252         ));
253         jScrollPane4.setViewportView(jTable2);
254 
255         jButton3.setText("Agregar");
256 
257         javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
258         jPanel2.setLayout(jPanel2Layout);
259         jPanel2Layout.setHorizontalGroup(
260             jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
261             .addComponent(jScrollPane4, javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE)
262             .addGroup(jPanel2Layout.createSequentialGroup()
263                 .addGap(39, 39, 39)
264                 .addComponent(jButton3)
265                 .addContainerGap(157, Short.MAX_VALUE))
266         );
267         jPanel2Layout.setVerticalGroup(
268             jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
269             .addGroup(jPanel2Layout.createSequentialGroup()
270                 .addComponent(jScrollPane4, javax.swing.GroupLayout.PREFERRED_SIZE, 115, javax.swing.GroupLayout.PREFERRED_SIZE)
271                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
272                 .addComponent(jButton3)
273                 .addGap(0, 10, Short.MAX_VALUE))
274         );
275 
276         jTabbedPane1.addTab("Comidas - ADD", jPanel2);
```

```
277
278         jButton1.setText("<--");
279         jButton1.addActionListener(new java.awt.event.ActionListener() {
280             public void actionPerformed(java.awt.event.ActionEvent evt) {
281                 jButton1ActionPerformed(evt);
282             }
283         });
284
285         jLabel5.setText("Cantidad:");
286
287         bSend.setText("Agente Dieta");
288         bSend.addActionListener(new java.awt.event.ActionListener() {
289             public void actionPerformed(java.awt.event.ActionEvent evt) {
290                 bSendActionPerformed(evt);
291             }
292         });
293
294         jTextArea1.setColumns(20);
295         jTextArea1.setRows(5);
296         jScrollPane3.setViewportView(jTextArea1);
297
298         jLabel6.setText("Recomendación Alimenticia:");
299
300         jLabel7.setText("Preferencias Alimenticias:");
301
302         jButton2.setText("-->");
303
304         javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
305         getContentPane().setLayout(layout);
306         layout.setHorizontalGroup(
307             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
308             .addGroup(layout.createSequentialGroup()
309                 .addContainerGap()
310                 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
311                     .addGroup(layout.createSequentialGroup()
312                         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
313                             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 0,
Short.MAX_VALUE)
314                         .addGroup(layout.createSequentialGroup()
315                             .addComponent(jLabel7)
316                             .addGap(0, 0, Short.MAX_VALUE)))
317                     .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
                .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 310,
318 javax.swing.GroupLayout.PREFERRED_SIZE)
319             .addContainerGap())
320           .addGroup(layout.createSequentialGroup()
321             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED_SIZE, 371,
322 javax.swing.GroupLayout.PREFERRED_SIZE)
323               .addComponent(jLabel6))
324             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
325             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
326               .addGroup(layout.createSequentialGroup()
327                 .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE, 66, Short.MAX_VALUE)
328                 .addGap(58, 58, 58))
329               .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
330                 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
331                   .addComponent(tCant, javax.swing.GroupLayout.Alignment.LEADING)
                  .addComponent(jButton2, javax.swing.GroupLayout.Alignment.LEADING,
332 javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                  .addComponent(jButton1, javax.swing.GroupLayout.Alignment.LEADING,
333 javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
334                 .addComponent(bSend, javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE))
335               .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)))
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
336 javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
337             .addGap(15, 15, 15))))
338     );
339     layout.setVerticalGroup(
340       layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
341       .addGroup(layout.createSequentialGroup()
342         .addContainerGap()
                .addComponent(jTabbedPane1, javax.swing.GroupLayout.PREFERRED_SIZE, 206,
343 javax.swing.GroupLayout.PREFERRED_SIZE)
344         .addGap(18, 18, 18)
345         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
346           .addGroup(layout.createSequentialGroup()
347             .addComponent(jLabel5)
348             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(tCant, javax.swing.GroupLayout.PREFERRED_SIZE,
349 javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
350             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
351             .addComponent(jButton2)
352             .addGap(0, 0, Short.MAX_VALUE))
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
353 javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
354         .addContainerGap())
355       .addGroup(layout.createSequentialGroup()
```

```
356             .addGap(8, 8, 8)
357             .addComponent(jLabel7)
358             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
359             .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 184,
    javax.swing.GroupLayout.PREFERRED_SIZE)
360             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
361             .addComponent(jLabel6)
362             .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
363             .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
364               .addComponent(jScrollPane3)
365               .addGroup(layout.createSequentialGroup()
366                 .addGap(70, 70, 70)
367                 .addComponent(jButton1)
368                 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
369                 .addComponent(bSend)
370                 .addGap(0, 0, Short.MAX_VALUE)))
371             .addGap(16, 16, 16))
372       );
373
374       pack();
375     }// </editor-fold>
376
377     private void jTree1MouseClicked(java.awt.event.MouseEvent evt) {
378       // TODO add your handling code here:
379       DefaultMutableTreeNode selectedElement
380       =(DefaultMutableTreeNode) jTree1.getSelectionPath().getLastPathComponent();
381       String sql1 = "SELECT idF, fDesc, fCarb, fProt, fGras, fEner, fGlu, fPot, fSod FROM fInfo WHERE fDesc ='"
382                 +selectedElement.getUserObject()+"'";
383       try {
384         st = cn1.createStatement();
385         rsP = st.executeQuery(sql1);
386         rsP.next();
387         tProt.setText(df.format(rsP.getDouble(4)).toString());
388         tCarbo.setText(df.format(rsP.getDouble(3)).toString());
389         tGras.setText(df.format(rsP.getDouble(5)).toString());
390         tEner.setText(df.format(rsP.getDouble(6)).toString());
391         cant = Double.parseDouble(tCant.getText());
392         info [0] = rsP.getString(1);
393         info [1] = rsP.getString(2);
394         info [2] = String.valueOf(rsP.getDouble(3));
395         info [3] = String.valueOf(rsP.getDouble(4));
396         info [4] = String.valueOf(rsP.getDouble(5));
```

```
397             info [5] = String.valueOf(rsP.getDouble(6));
398             info [6] = String.valueOf(rsP.getDouble(7));
399             info [7] = String.valueOf(rsP.getDouble(8));
400             info [8] = String.valueOf(rsP.getDouble(9));
401 //              System.out.print(info[0]+info[1]+info[2]+info[3]+info[4]+info[5]+info[6]+info[7]+info[8]);
402         } catch (SQLException ex) {
403             Logger.getLogger(cpDiet.class.getName()).log(Level.SEVERE, null, ex);
404         }
405     }
406 
407     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
408         // TODO add your handling code here:
409         if(cant!=0){
410             info [2] = String.valueOf(Double.parseDouble(info [2])*cant);
411             info [3] = String.valueOf(Double.parseDouble(info [3])*cant);
412             info [4] = String.valueOf(Double.parseDouble(info [4])*cant);
413             info [5] = String.valueOf(Double.parseDouble(info [5])*cant);
414             info [6] = String.valueOf(Double.parseDouble(info [6])*cant);
415             info [7] = String.valueOf(Double.parseDouble(info [7])*cant);
416             info [8] = String.valueOf(Double.parseDouble(info [8])*cant);
417             model.addRow(info);
418             foods.add(new foodN((String) info [0],
419             new String (info [1]),
420             new Double (info [2]),
421             new Double (info [3]),
422             new Double (info [4]),
423             new Double (info [5]),
424             new Double (info [6]),
425             new Double (info [7]),
426             new Double (info [8])));
427     }
428         else{
429             JOptionPane.showMessageDialog(null, "Introduzca la cantidad y de click al Alimento");
430         }
431     }
432 
433     private void bSendActionPerformed(java.awt.event.ActionEvent evt) {
434         // TODO add your handling code here:
435         String sDatAn = "";
436         String sDatFo = "";
437         String sDatUn = "";
```

```
438             for (int i=0; i<foods.size();i++) {
439                 if(i==foods.size()-1){
440                     sDatFo = sDatFo+foods.elementAt(i).getId()+"-"+foods.elementAt(i).getDesc()+"-"+
441                     foods.elementAt(i).getCarbo()+"-"+foods.elementAt(i).getProt()+"-"+
442                     foods.elementAt(i).getGras()+"-"+foods.elementAt(i).getEner()+"-"+
443                     foods.elementAt(i).getGlu()+"-"+foods.elementAt(i).getPot()+"-"+
444                     foods.elementAt(i).getSod();
445                 }
446                 else{
447                     sDatFo = sDatFo+foods.elementAt(i).getId()+"-"+foods.elementAt(i).getDesc()+"-"+
448                     foods.elementAt(i).getCarbo()+"-"+foods.elementAt(i).getProt()+"-"+
449                     foods.elementAt(i).getGras()+"-"+foods.elementAt(i).getEner()+"-"+
450                     foods.elementAt(i).getGlu()+"-"+foods.elementAt(i).getPot()+"-"+
451                     foods.elementAt(i).getSod()+"*";
452                 }
453 //              System.out.println(foods.size());
454             }
455         sDatAn = String.valueOf(df.format(vDatAna[0])+"/"+
456                     df.format(vDatAna[1])+"/"+df.format(vDatAna[2])+"/"+df.format(vDatAna[3])+"/"+
457                     df.format(vDatAna[4])+"/"+df.format(vDatAna[5])+"/"+df.format(vDatAna[6])+"/");
458 //      System.out.println(sDatFo);
459             sDatUn = sDatAn+sDatFo;
460             GuiEvent ge = new GuiEvent(this, SENT_TYPE);
461             ge.addParameter("Agent2");
462             ge.addParameter(sDatUn);
463             owner.postGuiEvent(ge);
464     }
465
466     /**
467      * @param args the command line arguments
468      */
469     public static void main(String args[]) {
470         /* Set the Nimbus look and feel */
471         //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
472         /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
473          * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
474          */
475         try {
476             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
477                 if ("Nimbus".equals(info.getName())) {
478                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
```

```
479                 break;
480             }
481         }
482     } catch (ClassNotFoundException ex) {
483         java.util.logging.Logger.getLogger(cpDiet.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
484     } catch (InstantiationException ex) {
485         java.util.logging.Logger.getLogger(cpDiet.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
486     } catch (IllegalAccessException ex) {
487         java.util.logging.Logger.getLogger(cpDiet.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
488     } catch (javax.swing.UnsupportedLookAndFeelException ex) {
489         java.util.logging.Logger.getLogger(cpDiet.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
490     }
491     //</editor-fold>
492 
493     /* Create and display the form */
494 }
495 
496 // Variables declaration - do not modify
497 private javax.swing.JButton bSend;
498 private javax.swing.JButton jButton1;
499 private javax.swing.JButton jButton2;
500 private javax.swing.JButton jButton3;
501 private javax.swing.JLabel jLabel1;
502 private javax.swing.JLabel jLabel2;
503 private javax.swing.JLabel jLabel3;
504 private javax.swing.JLabel jLabel4;
505 private javax.swing.JLabel jLabel5;
506 private javax.swing.JLabel jLabel6;
507 private javax.swing.JLabel jLabel7;
508 private javax.swing.JPanel jPanel1;
509 private javax.swing.JPanel jPanel2;
510 private javax.swing.JScrollPane jScrollPane1;
511 private javax.swing.JScrollPane jScrollPane2;
512 private javax.swing.JScrollPane jScrollPane3;
513 private javax.swing.JScrollPane jScrollPane4;
514 private javax.swing.JTabbedPane jTabbedPane1;
515 private javax.swing.JTable jTable1;
516 private javax.swing.JTable jTable2;
517 private javax.swing.JTextArea jTextArea1;
518 private javax.swing.JTree jTree1;
519 private javax.swing.JTextField tCant;
```

```
520     private javax.swing.JTextField tCarbo;
521     private javax.swing.JTextField tEner;
522     private javax.swing.JTextField tGras;
523     private javax.swing.JTextField tProt;
524     // End of variables declaration
525     String db = "jdbc:mysql://localhost/foodbC";
526     String db1 = "jdbc:mysql://localhost/Agentes";
527     public conectar cc = new conectar();
528     public Connection cn = cc.conexion(db);
529     public Connection cn1 = cc.conexion(db1);
530 }
```

A.1.2.4 Evaluación Antropométrica, Bioquímica y Nutricional.

El siguiente código pertenece a la clase análisis.java, desde esata clase se realizan las dos evaluaciónes antropométrica, bioquímica y nutricional.

```
1  package jade;
2
3  import jade.gui.GuiAgent;
4
5  /**
6   *
7   * @author omarlagunes
8   */
9  public class analisis {
10    float pes;
11    float tall;
12    float ed, iaf;
13    String ob;
14    String sex;
15    private GuiAgent owner;
16    public final int SENT_TYPE = 0;
17    double [] vDatAn = new double[7];
18
19    public analisis(float [] vDat, String sexo){
20        pes = vDat[0];
21        tall = vDat[1];
22        ed = vDat[4];
23        sex = sexo;
24        iaf = vDat[3];
25        calcular();
26     }
```

```
27    public String calcular(){
28        double IMC;
29        double Pid;
30        double RCT;
31        double carb;
32        double lip;
33        double prot;
34        double Sod;
35        IMC = pes/((double)Math.pow(tall, 2));
36        if (IMC>=19.9&&IMC<=24.9)ob="normal";
37        if (IMC>=25&&IMC<=29.9)ob="sobrepeso";
38        if (IMC>30)ob="Obesidad";
39        if (sex=="M"){
40          Pid = ((float)Math.pow(tall, 2))*23;
41          RCT = (10*pes)+(6.25f*(tall*100))-(5*ed)+5;
42          RCT = RCT * iaf;
43        }
44        else{
45          Pid = ((float)Math.pow(tall, 2))*21.5f;
46          RCT = (10*pes)+(6.25f*(tall*100))-(5*ed)-161;
47          RCT = RCT * iaf;
48        }
49
50        carb = (RCT*0.5f)/4;
51        prot = (RCT*0.2f)/4;
52        lip = (RCT*0.3f)/9;
53        Sod = (RCT/1000)*800;
54
55        vDatAn [0] = IMC;
56        vDatAn [1] = Pid;
57        vDatAn [2] = RCT;
58        vDatAn [3] = carb;
59        vDatAn [4] = prot;
60        vDatAn [5] = lip;
61        vDatAn [6] = Sod;
62
63        return Double.toString(IMC)+" "+ob+"\n"+"Peso Ideal: "+Double.toString(Pid)+"kg \n"+"RCT:
  "+Double.toString(RCT)+"kCal \n"
64              +"Carbohidratos: "+Double.toString(carb)+"gr \n"+"Proteinas: "+Double.toString(prot)+"gr \n"+
65               "Lipidos: "+Double.toString(lip)+"gr \n"+"Sodio: "+Double.toString(Sod)+"mg \n";
66    }
67  }
```

A.1.2.5 Alimento

El siguiente código pertenece a la clase foodN.java, esta clase es la encargada de crear un objeto que represente a un alimento, con su información nutricional, descripción e id.

```
1  package jade;
2
3  /**
4   *
5   * @author omarlagunes
6   */
7  public class foodN {
8      private String id;
9      private String Desc;
10     private double carbo;
11     private double prot;
12     private double gras;
13     private double ener;
14     private double glu;
15     private double pot;
16     private double sod;
17
18     public foodN (String id, String dec, Double Carb, Double Pro, Double Gras, Double Ene
19           , Double Glu, Double Pot, Double Sod){
20         this.id = id;
21         this.Desc = dec;
22         this.carbo = Carb;
23         this.prot = Pro;
24         this.gras = Gras;
25         this.ener = Ene;
26         this.glu = Glu;
27         this.pot = Pot;
28         this.sod = Sod;
29     }
30
31     /**
32      * @return the id
33      */
34     public String getId() {
35         return id;
36     }
37
```

```
38    /**
39     * @param id the id to set
40     */
41    public void setId(String id) {
42        this.id = id;
43    }
44
45    /**
46     * @return the Desc
47     */
48    public String getDesc() {
49        return Desc;
50    }
51
52    /**
53     * @param Desc the Desc to set
54     */
55    public void setDesc(String Desc) {
56        this.Desc = Desc;
57    }
58
59    /**
60     * @return the carbo
61     */
62    public double getCarbo() {
63        return carbo;
64    }
65
66    /**
67     * @param carbo the carbo to set
68     */
69    public void setCarbo(double carbo) {
70        this.carbo = carbo;
71    }
72
73    /**
74     * @return the prot
75     */
76    public double getProt() {
77        return prot;
78    }
```

```
79
80      /**
81       * @param prot the prot to set
82       */
83      public void setProt(double prot) {
84          this.prot = prot;
85      }
86
87      /**
88       * @return the gras
89       */
90      public double getGras() {
91          return gras;
92      }
93
94      /**
95       * @param gras the gras to set
96       */
97      public void setGras(double gras) {
98          this.gras = gras;
99      }
100
101     /**
102      * @return the ener
103      */
104     public double getEner() {
105         return ener;
106     }
107
108     /**
109      * @param ener the ener to set
110      */
111     public void setEner(double ener) {
112         this.ener = ener;
113     }
114
115     /**
116      * @return the glu
117      */
118     public double getGlu() {
119         return glu;
```

```
120     }
121
122     /**
123      * @param glu the glu to set
124      */
125     public void setGlu(double glu) {
126         this.glu = glu;
127     }
128
129     /**
130      * @return the pot
131      */
132     public double getPot() {
133         return pot;
134     }
135
136     /**
137      * @param pot the pot to set
138      */
139     public void setPot(double pot) {
140         this.pot = pot;
141     }
142
143     /**
144      * @return the sod
145      */
146     public double getSod() {
147         return sod;
148     }
149
150     /**
151      * @param sod the sod to set
152      */
153     public void setSod(double sod) {
154         this.sod = sod;
155     }
156 }
```

A.1.3 Agente Dieta

A.1.3.1 Clase Principal del Algoritmo genético.

El siguiente código pertenece a la clase DietCMain.java, es la parte principal del algoritmo genético, encargada de crear los genes y cromosomas que son las soluciones propuestas a la solicitud de una recomendación alimenticia.

```
1  package AgenteCal;
2  import java.io.*;
3  import org.jgap.*;
4  import org.jgap.data.*;
5  import org.jgap.impl.*;
6  import org.jgap.xml.*;
7  import org.w3c.dom.*;
8
9  public class DietCMain {
10
11   private final static String CVS_REVISION = "$Revision: 1.3 $";
12
13   private static final int MAX_ALLOWED_EVOLUTIONS = 600;
14
15   public static double[] itemVolumes;
16   public static double[] itemCarb;
17   public static double[] itemProt;
18   public static double[] itemLip;
19   public static String[] itemNames;
20
21   public static void findItemsForVolume(double a_knapsackVolume, double a_carb, double a_prot, double
     a_lip,
22         double []itemVolume, double []itemCar, double []itemPro, double []itemLi, String []itemName)
23     throws Exception {
24
25    System.out.print("-- Ejecutando AG --");
26
27    itemVolumes = itemVolume;
28    itemCarb = itemCar;
29    itemProt = itemPro;
30    itemLip = itemLi;
31    itemNames = itemName;
32
33    Configuration conf = new DefaultConfiguration();
34    conf.setPreservFittestIndividual(true);
35
36    FitnessFunction myFunc =
37       new DietCFitnessFunction(a_knapsackVolume, a_carb, a_prot, a_lip);
38    conf.setFitnessFunction(myFunc);
```

```
39    conf.addGeneticOperator(new MutationOperator(conf, 35));
40
41    Gene[] sampleGenes = new Gene[itemVolumes.length];
42    for (int i = 0; i < itemVolumes.length; i++) {
43      CompositeGene compositeGene = new CompositeGene(conf);
44
45      IntegerGene item = new IntegerGene(conf, 0,
46                              (int) Math.ceil(a_knapsackVolume /
47          itemVolumes[i]));
48      IntegerGene itemC = new IntegerGene(conf, 0,
49                              (int) Math.ceil(a_carb /
50          itemCarb[i]));
51      IntegerGene itemP = new IntegerGene(conf, 0,
52                              (int) Math.ceil(a_prot /
53          itemProt[i]));
54      IntegerGene itemL = new IntegerGene(conf, 0,
55                              (int) Math.ceil(a_prot /
56          itemLip[i]));
57
58      compositeGene.addGene(item);
59      compositeGene.addGene(itemC);
60      compositeGene.addGene(itemP);
61      compositeGene.addGene(itemL);
62      sampleGenes[i] = compositeGene;
63    }
64    IChromosome sampleChromosome = new Chromosome(conf, sampleGenes);
65    conf.setSampleChromosome(sampleChromosome);
66
67    conf.setPopulationSize(400);
68
69    Genotype population = Genotype.randomInitialGenotype(conf);
70
71    for (int i = 0; i < MAX_ALLOWED_EVOLUTIONS; i++) {
72      population.evolve();
73    }
74
75    DataTreeBuilder builder = DataTreeBuilder.getInstance();
76    IDataCreators doc2 = builder.representGenotypeAsDocument(population);
77
78    XMLDocumentBuilder docbuilder = new XMLDocumentBuilder();
79    Document xmlDoc = (Document) docbuilder.buildDocument(doc2);
```

```
80    XMLManager.writeFile(xmlDoc, new File("knapsackJGAP.xml"));
81
82    IChromosome bestSolutionSoFar = population.getFittestChromosome();
83    System.out.println("The best solution has a fitness value of " +
84                    bestSolutionSoFar.getFitnessValue());
85    System.out.println("It contained the following: ");
86    int count;
87    double totalVolume = 0.0d;
88    double totalCarb = 0.0d;
89    double totalProt = 0.0d;
90    double totalLipi = 0.0d;
91    for (int i = 0; i < bestSolutionSoFar.size(); i++) {
92      CompositeGene comp = (CompositeGene)bestSolutionSoFar.getGene(i);
93
94      IntegerGene item = (IntegerGene)comp.geneAt(0);
95      IntegerGene itemC = (IntegerGene)comp.geneAt(1);
96      IntegerGene itemP = (IntegerGene)comp.geneAt(2);
97      IntegerGene itemL = (IntegerGene)comp.geneAt(3);
98
99      count = ( (Integer) item.getAllele()).intValue();
100     if (count > 0) {
101
102       System.out.println("\t " + count + " x " + itemNames[i]);
103       totalVolume += itemVolumes[i] * count;
104       totalCarb += itemCarb[i] * count;
105       totalProt += itemProt[i] * count;
106       totalLipi += itemLip[i] * count;
107     }
108   }
109   System.out.println("\nFor a RCT of " + totalVolume + " kcal");
110   System.out.println("\nFor a total carbs of " + totalCarb + " gr");
111   System.out.println("\nFor a total prot of " + totalProt + " gr");
112   System.out.println("\nFor a total Lipi of " + totalLipi + " gr");
113   System.out.println("Expected volume was " + a_knapsackVolume + " gr");
114   System.out.println("\nRCT difference is " +
115                   Math.abs(totalVolume - a_knapsackVolume) + " kcal");
116   System.out.println("Expected carb was " + a_carb + " gr");
117   System.out.println("Volume difference is " +
118                   Math.abs(totalCarb - a_carb) + " gr");
119   System.out.println("Expected volume was " + a_prot + " gr");
120   System.out.println("Volume difference is " +
```

```
121                 Math.abs(totalProt - a_prot) + " gr");
122     System.out.println("Expected volume was " + a_lip + " gr");
123     System.out.println("Volume difference is " +
124                 Math.abs(totalLipi - a_lip) + " gr");
125   }
126 }
```

A.1.3.2 Función fitness.

El siguiente código pertenece a la clase DietCFitnessFunction.java, es la encargada de evaluar y obtener la mejor solución proporcionada por la clase DietCMain.java dependiendo los requerimentos nutriconales.

```
1  package AgenteCal;
2
3  import java.util.*;
4  import org.jgap.*;
5  import org.jgap.impl.*;
6
7  public class DietCFitnessFunction
8     extends FitnessFunction {
9
10   private final static String CVS_REVISION = "$Revision: 1.2 $";
11
12   private final double m_knapsackVolume;
13   private final double m_Carb;
14   private final double m_Prot;
15   private final double m_Lipi;
16
17   public static final double MAX_BOUND = 1000000.0;
18
19   private static final double ZERO_DIFFERENCE_FITNESS = Math.sqrt(MAX_BOUND);
20
21   public DietCFitnessFunction(double a_knapsackVolume, double a_carb, double a_prot, double a_lipi) {
22     if (a_knapsackVolume < 1 || a_knapsackVolume >= MAX_BOUND) {
23       throw new IllegalArgumentException(
24           "Knapsack volumen must be between 1 and " + MAX_BOUND + ".");
25     }
26     m_knapsackVolume = a_knapsackVolume;
27     m_Carb = a_carb;
28     m_Prot = a_prot;
29     m_Lipi = a_lipi;
30   }
31
```

```
32   public double evaluate(IChromosome a_subject) {
33
34     double totalVolume = getTotalVolume(a_subject);
35     double totalCarbs = getTotalCarb(a_subject);
36     double totalProte = getTotalProt(a_subject);
37     double totalLipid = getTotalLipi(a_subject);
38     int numberOfItems = getTotalNumberOfItems(a_subject);
39  //    System.out.println(numberOfItems);
40     double volumeDifference = Math.abs(m_knapsackVolume - totalVolume);
41     double volumeDifferenceC = Math.abs(m_Carb - totalCarbs);
42     double volumeDifferenceP = Math.abs(m_Prot - totalProte);
43     double volumeDifferenceL = Math.abs(m_Lipi - totalLipid);
44     double fitness = 0.0d;
45
46     fitness += volumeDifferenceC(MAX_BOUND, volumeDifferenceC)+
47           volumeDifferenceP(MAX_BOUND, volumeDifferenceP)+
48           volumeDifferenceL(MAX_BOUND, volumeDifferenceL)+
49           volumeDifferenceBonus(m_knapsackVolume, volumeDifference);
50
51     fitness -= computeItemNumberPenalty(a_subject, MAX_BOUND, numberOfItems);
52
53     return Math.max(1.0d, fitness);
54   }
55
56
57   protected double volumeDifferenceBonus(double a_maxFitness,
58                                 double a_volumeDifference) {
59     if (a_volumeDifference == 0) {
60       return a_maxFitness;
61     }
62     else {
63
64       return a_maxFitness - (a_volumeDifference);
65     }
66   }
67
68   protected double volumeDifferenceC(double a_maxFitness,
69                                 double a_volumeDifferenceC) {
70     if (a_volumeDifferenceC == 0) {
71       return a_maxFitness;
72     }
```

```
73    else {
74
75      return a_maxFitness - (a_volumeDifferenceC);
76    }
77  }
78
79   protected double volumeDifferenceP(double a_maxFitness,
80                                      double a_volumeDifferenceP) {
81    if (a_volumeDifferenceP == 0) {
82      return a_maxFitness;
83    }
84    else {
85
86      return a_maxFitness - (a_volumeDifferenceP);
87    }
88  }
89
90   protected double volumeDifferenceL(double a_maxFitness,
91                                      double a_volumeDifferenceL) {
92    if (a_volumeDifferenceL == 0) {
93      return a_maxFitness;
94    }
95    else {
96
97      return a_maxFitness - (a_volumeDifferenceL);
98    }
99  }
100
101  protected double computeItemNumberPenalty(IChromosome a_potentialSolution,
102                                      double a_maxFitness, int a_items) {
103   if (a_items == 0) {
104
105     return 0;
106   }
107   else {
108
109     double penalty = (Math.min(a_maxFitness, a_items));
110     return Math.min(a_maxFitness, penalty);
111   }
112 }
113
```

```
114   public static double getTotalVolume(IChromosome a_potentialSolution) {
115     double volume = 0.0d;
116     for (int i = 0; i < a_potentialSolution.size(); i++) {
117       CompositeGene comp = (CompositeGene)a_potentialSolution.getGene(i);
118       volume += getNumberOfItemsAtGene(comp) * DietCMain.itemVolumes[i];
119 //      System.out.println(a_potentialSolution+" "+comp+" "+volume+" "+i);
120     }
121     return volume;
122   }
123
124   public static double getTotalCarb(IChromosome a_potentialSolution) {
125     double carb = 0.0d;
126     for (int i = 0; i < a_potentialSolution.size(); i++) {
127       CompositeGene comp = (CompositeGene)a_potentialSolution.getGene(i);
128       carb += getNumberOfItemsAtGene(comp) * DietCMain.itemCarb[i];
129 //      System.out.println(a_potentialSolution+" "+comp+" "+volume+" "+i);
130     }
131     return carb;
132   }
133
134   public static double getTotalProt(IChromosome a_potentialSolution) {
135     double prot = 0.0d;
136     for (int i = 0; i < a_potentialSolution.size(); i++) {
137       CompositeGene comp = (CompositeGene)a_potentialSolution.getGene(i);
138       prot += getNumberOfItemsAtGene(comp) * DietCMain.itemProt[i];
139 //      System.out.println(a_potentialSolution+" "+comp+" "+volume+" "+i);
140     }
141     return prot;
142   }
143
144   public static double getTotalLipi(IChromosome a_potentialSolution) {
145     double lipi = 0.0d;
146     for (int i = 0; i < a_potentialSolution.size(); i++) {
147       CompositeGene comp = (CompositeGene)a_potentialSolution.getGene(i);
148       lipi += getNumberOfItemsAtGene(comp) * DietCMain.itemLip[i];
149 //      System.out.println(a_potentialSolution+" "+comp+" "+volume+" "+i);
150     }
151     return lipi;
152   }
153
154   public static int getNumberOfItemsAtGene(CompositeGene a_compositeGene) {
```

```
155     Integer numItems =
156       (Integer) a_compositeGene.geneAt(0).getAllele();
157     return numItems.intValue();
158   }
159
160   public static int getTotalNumberOfItems(IChromosome a_potentialSolution) {
161     int totalItems = 0;
162     int numberOfGenes = a_potentialSolution.size();
163     for (int i = 0; i < numberOfGenes; i++) {
164       CompositeGene comp = (CompositeGene)a_potentialSolution.getGene(i);
165       totalItems += getNumberOfItemsAtGene(comp);
166     }
167     return totalItems;
168   }
169 }
```

A.1.4 Agente Evaluación

A.1.4.1 Control difuso.

El siguiente código pertenece al archivo de Xfuzzy 3, agenteEvaluador.xfl, contiene los parámetros que componen el control difuso encargado de evaluar la efectividad de la recomendación alimenticia proporcionada.

```
1   ortype Aglucosa [-120.0,120.0;61] {
2     muyMala xfl.trapezoid(-160.0,-120.0,-80.0,-60.0);
3     buena xfl.triangle(-90.0,-45.0,0.0);
4     normal xfl.triangle(-50.0,0.0,50.0);
5     muybuena xfl.triangle(0.0,45.0,90.0);
6     mala xfl.trapezoid(60.0,80.0,120.0,160.0);
7   }
8
9   type Apresion [-80.0,80.0;61] {
10    mala xfl.trapezoid(-106.66666666666667,-80.0,-53.33333333333333,-40.0);
11    buena xfl.triangle(-50.0,-25.0,0.0);
12    normal xfl.triangle(-20.0,0.0,20.0);
13    muybuena xfl.triangle(0.0,25.0,50.0);
14    muymala xfl.trapezoid(40.0,53.33333333333333,80.0,106.66666666666667);
15  }
16
17  type Evaluacion [-200.0,200.0;61] {
18    muymala xfl.trapezoid(-266.6666666666667,-200.0,-133.33333333333331,-
66.66666666666666);
19    mala xfl.triangle(-133.33333333333331,-66.66666666666666,0.0);
20    normal xfl.triangle(-66.66666666666666,0.0,66.66666666666669);
```

```
21   buena xfl.triangle(0.0,66.66666666666669,133.33333333333337);
22   muybuena xfl.trapezoid(66.66666666666666,133.33333333333331,200.0,266.6666666666667);
23   }
24
25   rulebase Reglas (Aglucosa FacRegGlucosa, Apresion FacRegPresion : Evaluacion tEvaluacion) {
26   if(FacRegGlucosa == muyMala & FacRegPresion == mala) -> tEvaluacion = mala;
27   if(FacRegGlucosa == muyMala & FacRegPresion == buena) -> tEvaluacion = mala;
28   if(FacRegGlucosa == muyMala & FacRegPresion == normal) -> tEvaluacion = mala;
29   if(FacRegGlucosa == muyMala & FacRegPresion == muybuena) -> tEvaluacion = mala;
30   if(FacRegGlucosa == muyMala & FacRegPresion == muymala) -> tEvaluacion = muymala;
31   if(FacRegGlucosa == buena & FacRegPresion == mala) -> tEvaluacion = mala;
32   if(FacRegGlucosa == buena & FacRegPresion == buena) -> tEvaluacion = buena;
33   if(FacRegGlucosa == buena & FacRegPresion == normal) -> tEvaluacion = buena;
34   if(FacRegGlucosa == buena & FacRegPresion == muybuena) -> tEvaluacion = muybuena;
35   if(FacRegGlucosa == buena & FacRegPresion == muymala) -> tEvaluacion = mala;
36   if(FacRegGlucosa == normal & FacRegPresion == mala) -> tEvaluacion = mala;
37   if(FacRegGlucosa == normal & FacRegPresion == buena) -> tEvaluacion = buena;
38   if(FacRegGlucosa == normal & FacRegPresion == normal) -> tEvaluacion = normal;
39   if(FacRegGlucosa == normal & FacRegPresion == muybuena) -> tEvaluacion = buena;
40   if(FacRegGlucosa == normal & FacRegPresion == muymala) -> tEvaluacion = mala;
41   if(FacRegGlucosa == muybuena & FacRegPresion == mala) -> tEvaluacion = normal;
42   if(FacRegGlucosa == muybuena & FacRegPresion == buena) -> tEvaluacion = muybuena;
43   if(FacRegGlucosa == muybuena & FacRegPresion == normal) -> tEvaluacion = muybuena;
44   if(FacRegGlucosa == muybuena & FacRegPresion == muybuena) -> tEvaluacion = muybuena;
45   if(FacRegGlucosa == muybuena & FacRegPresion == muymala) -> tEvaluacion = normal;
46   if(FacRegGlucosa == mala & FacRegPresion == mala) -> tEvaluacion = mala;
47   if(FacRegGlucosa == mala & FacRegPresion == buena) -> tEvaluacion = normal;
48   if(FacRegGlucosa == mala & FacRegPresion == normal) -> tEvaluacion = normal;
49   if(FacRegGlucosa == mala & FacRegPresion == muybuena) -> tEvaluacion = normal;
50   if(FacRegGlucosa == mala & FacRegPresion == muymala) -> tEvaluacion = muymala;
51   }
52
53   system (Aglucosa FacRegGlucosa, Apresion FacRegPresion : Evaluacion tEvaluacion) {
54   Reglas(FacRegGlucosa, FacRegPresion : tEvaluacion);
55   }
```