FISI LOGIC: Una nueva herramienta para sistemas de inferencia borroso

Rolando Maguiña Pérez[1], Sheila Campos Briceño[2], Daniel Carpio Contreras[2] y Joel Flores Martínez[2]

[1]Facultad de Ingeniería de Sistemas e Informática
Universidad Nacional mayor de San Marcos

rolando_maguina@yahoo.com

RESUMEN

El presente artículo tiene como objetivo presentar el sistema software denominado FISI Logic, desarrollado por los alumnos del curso Sistemas Inteligentes de la Facultad de Ingeniería de Sistemas e Informática de la UNMSM durante el semestre 2009-2 bajo la supervisión del profesor del curso. El proyecto consistió en el diseño y la construcción de una herramienta computacional de propósito general y de licencia libre para implementar Sistemas de Inferencia Borroso (SIB), obteniéndose como resultado la primera versión del aplicativo (versión alfa), la que a su vez servirá de base para futuros desarrollos. FISI Logic permite el modelado de sistemas Mamdani y Sugeno con sus respectivos métodos de inferencia borrosa. Su estructura está compuesta por cuatro módulos (Creación de variables de entrada y salida, Creación de la base de reglas borrosas y borrosificación, Mecanismo de Inferencia y Desborrosificación), integrados mediante una interfaz de usuario amigable; permite la resolución de problemas con variables cuyo universo es continuo. El sistema software ha sido implementado en la plataforma Java y MySQL como gestor de Base de Datos. Con resultados contrastados con el caso de estudio presentado en [19], se validó el sistema FISI Logic.

Palabra clave:

ABSTRACT

This article aims to present the software system called FISI Logic, developed by Intelligent Systems course students of the Faculty of Engineering and Computer Systems of San Marcos University during the 2009-2 semester under the supervision of course teacher. The project involved the design and construction of a general purpose computer tool and free license to implement Fuzzy Inference Systems, resulting in the first version of the application (version alpha), which in turn serve as basis for future development. FISI Logic enables modeling Mamdani and Sugeno systems with their fuzzy inference methods, and their structure consists of four modules (Creation of input and output variables, creation of the fuzzy rule base and borrosification, Inference Mechanism and Desborrosification), integrated with a user-friendly interface; allows solving problems with variables whose universe is continuous. The software system has been implemented in the Java platform and MySQL as database manager. With results contrasted with the case study presented in [19], the system FISI Logic was validated.

Keywords:

INTRODUCCIÓN

En los últimos años la lógica difusa ha tenido una gran aceptación y ha sido adoptada en una gran variedad de aplicaciones. La lógica borrosa, que se basa en los conjuntos borrosos introducidos por Zadeh [1], emula el razonamiento aproximado en lugar del preciso. Usa reglas heurísticas de la forma SI (antecedente) ENTONCES (consecuente), donde el antecedente y el consecuente son también conjuntos borrosos, ya sea puros o el resultado de operar con ellos. La certeza con que a una variable x se le puede asignar un valor lingüístico (conjunto borroso) se indica por una función de pertenencia.

La lógica borrosa puede modelizar y funcionar con expresiones que se usan en el cotidiano tales como "hace mucho calor", "no es muy alto", "el ritmo del corazón está un poco acelerado", etc. Esto hace posible incorporar el conocimiento de un experto en un dominio específico a través de valores lingüísticos.

Para resolver problemas mediante sistemas basados en lógica difusa existe una gama de herramientas y librerías tales como fuzzyTech [2] (de la empresa INFORM GmbH), Fuzzy Logic Toolbox, una caja de herramientas de la plataforma Matlab [3] (de la empresa MathWorks), y jFuzzyLogic [4]; éste es un paquete basado en Java que permite la selección de distintas funciones de pertenencia, métodos de desborrosificación y de implicación, y las dos primeras son las herramientas de carácter general de uso más extendido, tanto en el mundo académico como el profesional [5].

"Sin embargo, existen otras muchas herramientas útiles a la hora de desarrollar estos sistemas. FuzzyCLIPS [6] (de Togai Infralogic) es un paquete software que proporciona métodos para el diseño, la depuración y el test de sistemas expertos basados en lógica borrosa. FIDE [7] (de Aptronix) es un conjunto de herramientas para facilitar el desarrollo de productos basados en lógica borrosa integrado con microcontroladores comerciales, ofreciendo ayudas para la depuración, la simulación y control en tiempo real. El entorno Xfuzzy [8] es una plataforma de desarrollo de sistemas difusos complejos que facilitan las diferentes etapas de diseño desde su descripción inicial hasta la implementación final" [5].

Si bien ya existen entornos de desarrollo en el mercado, las de mayor aceptación se encuentran bajo licencia de pago y las de licencia libre poseen características que algunas veces no se ajustan a las necesidades de los usuarios. Por otra parte teníamos desde hace un buen tiempo la idea de desarrollar un trabajo de investigación a pequeña escala con la participación de los alumnos de alguno de los cursos. Específicamente el objetivo era la construcción de una herramienta de carácter general para implementar algún sistema inteligente. Surgió entonces la oportunidad de encargar esta tarea a los alumnos del curso denominado Sistemas Inteligentes correspondiente al semestre 2009-2, de la Facultad de Ingeniería de Sistemas e Informática – UNMSM como proyecto computacional del curso. La hipótesis planteada bajo un enfoque constructivista de la educación, era que si los alumnos del mencionado curso participaban en las etapas de análisis, diseño y desarrollo de sistema software propuesto, se produciría un mejor aprendizaje de esta técnica de la inteligencia computacional especialmente en sus aspectos prácticos. Más aún, la obtención de un prototipo serviría como un valioso medio de aprendizaje para los alumnos del curso y como punto de partida para futuros desarrollos.

Las herramientas privativas, descritas brevemente en la primera parte de este capítulo, han servido de inspiración para el sistema software denominado FISI Logic, cuya implementación es presentada en el capítulo III. En el capítulo II se presentan sucintamente los fundamentos de los sistemas de inferencia borrosos y de la metodología de desarrollo denominada Programación Extrema. En el capítulo IV se describen los experimentos computacionales requeridos para validar el sistema y, finalmente, en el V se presentan las conclusiones del presente trabajo.

FUNDAMENTACIÓN TEÓRICA

Sistema de Inferencia Borroso (SIB)

Son sistemas expertos con razonamiento aproximado que mapean un vector de entradas a una salida única (escalar). Se basan en la lógica borrosa para efectuar ese mapeo.

En la figura se muestra la arquitectura de un SIB mostrando los módulos que lo conforman y la forma en que se relacionan. Mayor información sobre los sistemas de inferencia basados en lógica borrosa y sobre lógica borrosa en general, se puede encontrar en [9], [10] y [11].

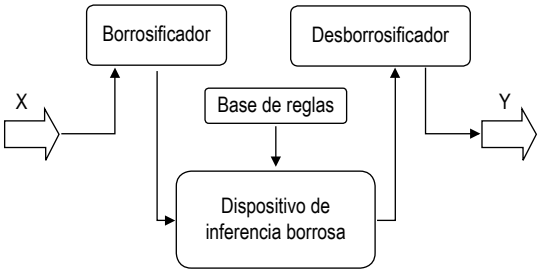


Figura N.º 1. Estructura de un SIB.

Metodologías de Desarrollo

Son un marco de trabajo (framework) usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información [12]. A lo largo del tiempo, una gran cantidad de métodos han sido desarrollados diferenciándose por su fortaleza y debilidad [13].

El framework para metodología de desarrollo de software consiste en [13]:

- Una filosofía de desarrollo de software con el enfoque del proceso de desarrollo de software.
- Herramientas, modelos y métodos para asistir al proceso de desarrollo de software.

Metodologías Agiles

Según lo indicado en [14] "Las metodologías ágiles intentan reducir el riesgo y elevar la productividad desarrollando el software en pequeñas iteraciones o miniversiones. Para eso evitan los tortuosos y burocráticos caminos de las metodologías tradicionales enfocándose en la gente y los resultados. Promueve iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto.

Programación Extrema

En [15] se define así: "La Programación Extrema es una de las metodologías de desarrollo más extendidas de entre las conocidas como metodologías ágiles. La idea principal es la optimización del tiempo de desarrollo en relación con los recursos empleados, mejorando la productividad y sin perder calidad en las implementaciones, e incluso incluyendo medidas para la mejora de la misma"

Los valores de la Programación Extrema (Extreme Programming o XP) son [16]:

- **Comunicación:** La Programación Extrema busca técnicas que fomenten una comunicación informal, pero intensa sincera y constante;
- **Simplicidad:** XP propone el principio de hacer la cosa más simple que pueda funcionar, en relación al proceso y la codificación. Es mejor hacer hoy algo simple, que hacerlo complicado y probablemente nunca usarlo mañana.
- **Retroalimentación** (o feedback): Nos indica si un trabajo está bien hecho. La Programación Extrema busca técnicas para tener feedback rápido y frecuente, para saber si algo está bien o mal hecho nada más terminarlo.
- **Coraje** (valor): Muchas acciones requieren coraje, dar abierta y sinceramente una opinión, por ejemplo, demanda valentía; si queremos un código sencillo tendremos que tirar código.

IMPLEMENTACIÓN DE FISI LOGIC

Los equipos de trabajo conformados por los alumnos del curso y supervisados por el profesor del mismo, tuvieron a su cargo la construcción de los módulos que constituyen un sistema de inferencia borroso y de la interfaz que los integra.

Enseguida se presentan las fases que demandó la construcción del sistema software FISI Logic. Esta implementación se basó en las clases impartidas en el curso, complementada por varias de las referencias enumeradas en este artículo, y en el conocimiento obtenido por los alumnos en los cursos relacionados, durante los semestres de estudio en la facultad.

Metodologías de Desarrollo

Se utilizó la Programación Extrema (XP) como metodología de desarrollo. Como sugiere XP se utilizó un modelo de trabajo usando repositorios de código donde los programadores publican sus códigos implementados y corregidos

Prácticas utilizadas

A. Planificación

a) **Equipo completo:** Formado por los alumnos (equipos de trabajo) y el profesor del curso (Jefe del Proyecto).
   - Equipo de Coordinación
   - Equipo de Creación de Variables
   - Equipo de Base de Reglas Borrosas
   - Equipo de Inferencia

- Equipo de Desborrosificación
- Jefe del Proyecto

b) Historias de usuario

i. Análisis de la creación de un SIB:

- Definir los objetivos, las restricciones y módulos del Sistema.
- Identificar las variables de entrada y salida: Variables y sus términos lingüísticos.

ii. Diseño de un SIB:

- Definir los conjuntos borrosos de cada etiqueta lingüística.
- Definir las reglas borrosas
- Definir el marco de los procedimientos empleados (Inferencia).
- Especificar la forma requerida de la salida del sistema (Desborrosificación).
- Ejecutar pruebas para validar el sistema.
- Verificar si la solución es compatible con el Análisis (i):

c) Planificación: El Jefe del Proyecto elaboró el cronograma de entrega de las miniversiones y la integración final. Se desarrolló a partir de la tercera semana de iniciado el semestre con un tiempo de duración de once semanas.

Miniversión o Módulo	Fecha
Creación de Variables	6ta semana de clases
Base de Reglas Borrosas	9na semana de clases
Mecanismo de Inferencia	11va semana de clases
Desborrosificación	13va semana de clases

Y la integración final se presentó en la última semana de clases.

d) Reuniones Continuas: Comunicación diaria entre los miembros de cada.

e) Versiones pequeñas: Se dividió en cuatro miniversiones (Creación de variables, Base de Reglas Borrosas, Mecanismo de Inferencia y Desborrosificación)., cada una de las cuales fue encargada a los equipos de trabajo.

f) Repositorio: XP-Dev.com

B. Diseño y Codificación

a) Diseño simple: Se buscó la sencillez en la programación así como la estandarización de las entidades a usar. Se diseñaron los diagramas de clases de todos los módulos desarrollados, además de los diagramas presentados a continuación:

- Diagrama de Casos de Uso del Sistema

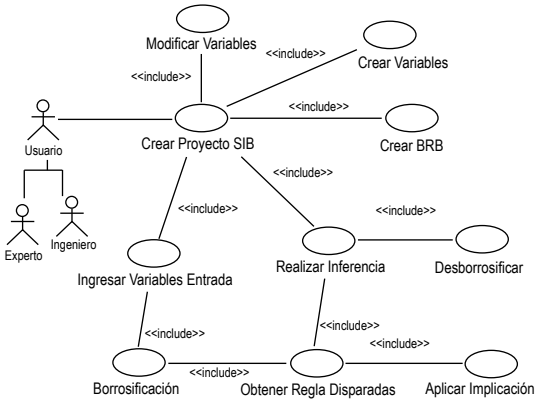


Figura N.º 2. Diagrama de Casos de Uso.

- Diagrama de Paquetes del sistema.

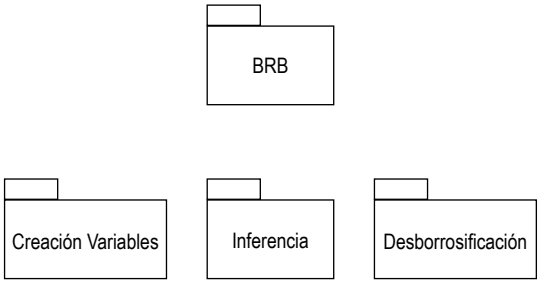


Figura N.º 3. Diagrama de Paquetes.

- Diagrama de actividades.
  (Ver figura N.º 4.)
- Diagrama de la Base de Datos
  (Ver figura N.º 5.)

b) Mejora del diseño: Al ir codificándose se mejoró tanto el código como los prototipos iniciales de las interfaces; además de extraer funcionalidades comunes, eliminar líneas de código innecesarias, etc.

c) Integración continua: Los responsables de cada grupo en conexión con el grupo coordinador se hicieron cargo de la actualización del proyecto (código fuente) que se encontraba en el repositorio.

d) El código es de todos: Cualquiera podía conocer y editar cualquier parte del código.

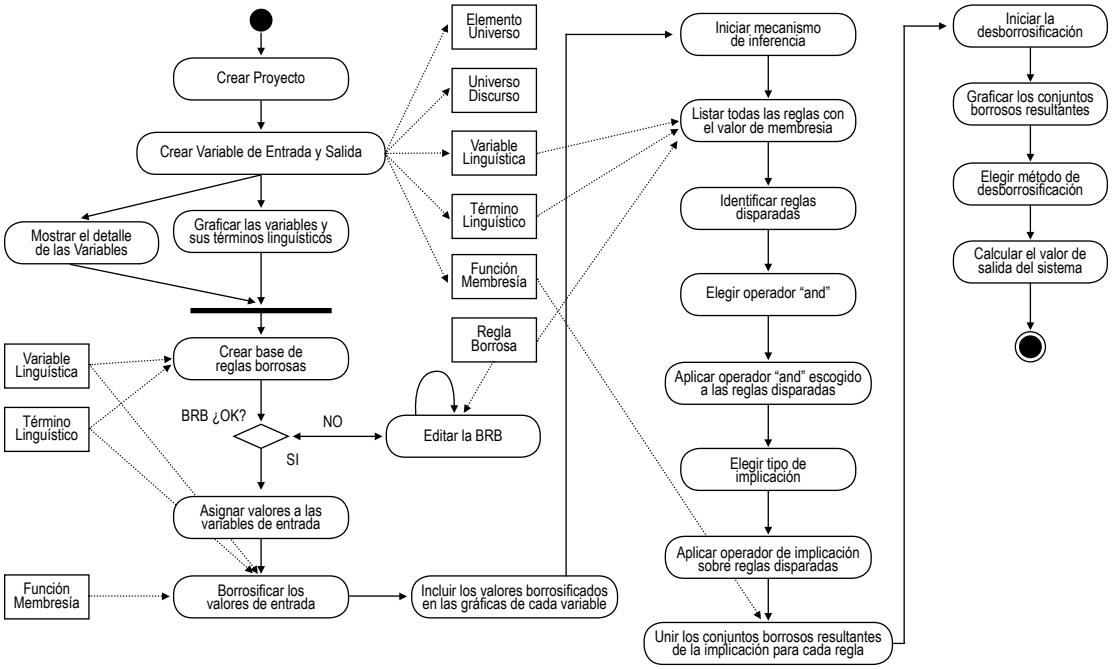


Figura N.º 4. Diagrama de actividades.

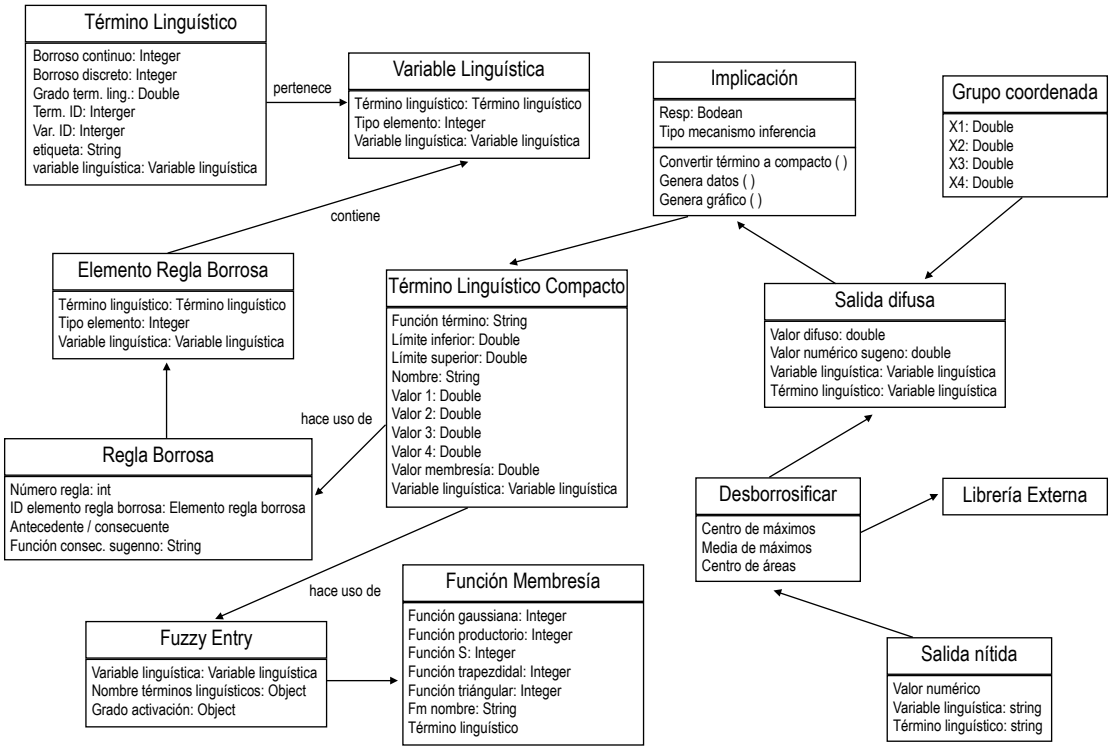


Figura N.º 5. Diagrama de la Base de Datos.

e) Normas de codificación:

- Estandarización de variables
- Estandarización de interfaces
- El Jefe de Proyecto definió las entradas y salidas para cada mini-versión.
- Los responsables de cada grupo definieron las estructuras de las entradas y las salidas de sus mini-versiones.

f) Metáforas: El Jefe de Proyecto definió frases o nombres que especificaban el funcionamiento de las distintas partes del programa, de forma que los equipos podían tener una mejor idea a partir de esos nombres, de dicho funcionamiento.

- Borrosificador
- Inferencia borrosa
- Desborrosificador
- Membresía
- Disparo de reglas
- Implicación borrosa
- Agregación

C. Prueba

a) Pruebas modulares: Basadas en ejemplos presentados en clases.

b) Desarrollo guiado por pruebas automáticas: Cada módulo usó las variables de entrada y salida (definidas en el análisis) como parámetros de las pruebas automáticas.

3.2. Mini-Versiones o Módulos

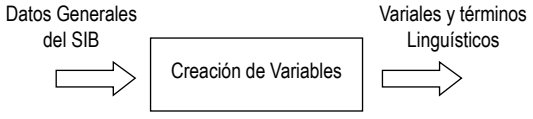
3.2.1 Módulo de creación de las variables de entrada y de salida del sistema.

Este módulo permite la creación de las variables de entrada y salida del sistema a través de una interfaz. Con este módulo el usuario podrá definir los parámetros de cada variable: nombre de la variable, universo de discurso, granularidad, los términos lingüísticos con sus respectivas funciones de pertenencia, etc. Este módulo dispone de las siguientes funciones de pertenencia: triangular, trapezoidal, función S, función ∏ y la función gaussiana.

Las características más importantes de este módulo son:

- Permite crear un proyecto.
- Las variables lingüísticas son definidas según criterio del usuario del sistema.
- Se debe especificar un nombre y un tipo de universo de discurso con sus respectivos valores (rango o conjunto de elementos según el universo de discurso sea continuo o discreto). Además, se deben especificar los términos lingüísticos (cantidad y etiqueta de cada término).
- Por cada término lingüístico se debe especificar su función de pertenencia indicando para cada función los valores de sus parámetros.
- El sistema cuenta con un mecanismo de validación de los parámetros de manera que éstos sean ingresados correctamente. Los cambios se reflejan inmediatamente en la vista previa de los términos lingüísticos, generada a partir de los valores especificados.
- Por último y de manera opcional es posible ingresar una medida y una descripción de la variable creada.
- Se puede modificar el nombre y el universo de discurso de las variables.
- Es posible eliminar alguna de las variables.
- Se puede mostrar por cada variable, haciendo uso de la librería JFreeChart [17], una gráfica representando su universo discurso y las funciones de pertenencia asociadas a cada uno de los términos lingüísticos de la variable actualmente seleccionada.

A. Entradas y salidas del módulo



3.2.2 Módulo de creación de la base de reglas borrosas (BRB) y de la borrosificación.

Este módulo permite la creación de la base de reglas borrosas proporcionando al usuario una herramienta que le brinde facilidades para el ingreso de la base de conocimiento ya sea mediante reglas en formato Mandami o en formato Sugeno.

Las características más importantes de esta parte del módulo son:

- Se identifican los términos lingüísticos tanto de las variables de entrada como los de la salida. Las entradas unidas por un conector lógico, por defecto

AND, constituyen los antecedentes, y las salidas, los consecuentes.

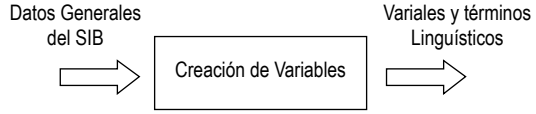
- Se combinan en el formato: SI antecedentes ENTONCES consecuentes.
- Los antecedentes y consecuentes se pueden modificar.
- Se pueden eliminar las reglas.
- Se puede modificar las salidas de las reglas en formato Sugeno. Ejm. Salida1*2 +4
- Las reglas se presentan en una tabla dinámica.

Además el módulo permite la borrosificación de los valores de las variables de entrada, actualizando este valor del atributo en el respectivo término lingüístico. Además inserta tanto el valor de entrada (nítido) como los valores de pertenencia resultantes, en la gráfica de cada variable de entrada.

Las características más importantes de esta parte son:

- Se captura el array de términos lingüísticos perteneciente a cada variable. Ejm. Etiqueta1, Etiqueta2, Etiqueta3.
- Se hace un CAST a partir de la función de pertenencia (atributo del término lingüístico) para identificar cuál es la función a la que pertenece el término (triangular, trapezoidal, etc.).
- Una vez identificada la función se evalúa en ésta el valor nítido, es decir, la entrada es el valor del eje de las abscisas X y la función nos devuelve el valor de las ordenadas Y. Finalmente el valor de membresía Y, se asigna como atributo al término lingüístico.
- Grafica el valor nítido y los valores de pertenencia para todas las variables, valiéndose de la librería JFreeChart.

A. Entradas y salidas del módulo



3.2.3 Módulo de mecanismo de inferencia borrosa

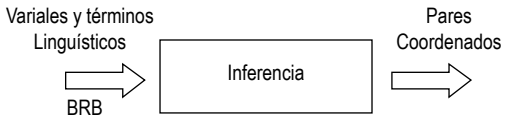
Obtenidas la BRB y los términos lingüísticos activados en el lado izquierdo de las reglas con sus respectivos grados de membresía, se procede a realizar la inferencia determinándose las reglas disparadas, y los términos lingüísticos de la Variable de Salida activada para cada regla, y sus respectivos grados de pertenencia.

Las características más importantes de esta parte del módulo son:

- Se puede seleccionar el operador de implicación entre los operadores Mamdani, Larsen y Zadeh.
- El módulo se conecta con la tabla 'REGLA_BORROSA' y lee todas las reglas.
- Se filtra las reglas que tienen como antecedentes los términos lingüísticos con grado de membresía mayor que cero, en otras palabras las reglas que se dispararon.
- A partir del operador seleccionado y las reglas disparadas, se aplica el operador de implicación seleccionado obteniéndose los consecuentes resultantes (términos lingüísticos) y sus respectivos grados de membresía
- En el caso de los sistemas Mamdani:
  - Los consecuentes son los términos lingüísticos definidos en las reglas borrosas.
  - A partir de los conjuntos borrosos asociados a los consecuentes resultantes de la inferencia y sus respectivos grados de membresía, se determinan los puntos en que sus correspondientes funciones de pertenencia son interceptadas. Una lista de estos puntos, llamados pares coordenados, constituyen la salida del módulo. Este procedimiento en la práctica, determina "el conjunto borroso global de salida que está dado por la unión o agregación de los conjuntos borrosos resultantes de la implicación para cada regla" [10].
  - Se grafica el conjunto borroso global de salidas resultantes valiéndose de la librería JFreeChart pero sin sus grados de membresía.
- En el caso de los sistemas Sugeno:
  - Los consecuentes son las funciones definidas en la base de reglas borrosas, éstas son evaluadas calculando su valor F(entradas).
  - El resultado final del sistema se obtiene del cálculo de:

$$\frac{\Sigma\ F(\text{entradas}) * \text{Valores de membresía}}{\Sigma\ \text{Valores de membresía}}$$

A. Entradas y salidas del módulo



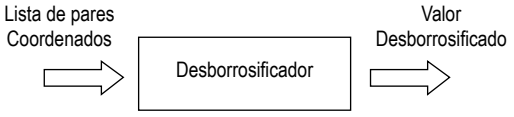
3.2.4 Módulo de desborrosificación

Es un mapeador de salida que convierte el conjunto borroso de salida, a una salida nítida.

Las características más importantes de esta parte son:

- Se grafica la unión o agregación de los pares coordenados.
- Se elige uno de los métodos considerados, verbigracia: Suma de Centro de Áreas (SCOA), Centro de Máximos (CoM), y Media de Máximos (MoM).
- El método Suma de Centro de Áreas es el más preciso pero su desventaja es la complejidad para hallar el valor de las integrales definidas. Una solución para resolver esta dificultad es utilizar métodos numéricos para aproximarlas. Para este proceso se usó el método denominado cuadratura de Gauss.
- Se calcula el valor desborrosificado a través de la librería científica basada en Java denominada flanagan.jar [18].

A. Entradas y salidas del módulo



4. EXPERIMENTOS COMPUTACIONALES

Requerimientos Mínimos

Hardware Sistema Operativo: Windows 98/2000/XP/7, Linux

Velocidad del procesador: 1 GHz

Capacidad de memoria: 512 MB

Software Componente necesario: Java Runtime Environment JRE 6.x

Base de datos local: MySQL

V. RESULTADOS

Procederemos a demostrar el uso del sistema software FISI Logic mediante el desarrollo de un ejemplo clásico de la lógica borrosa, presentado en [19], en el cual se define el diseño del SIB (variables de entrada y salida, base de reglas borrosas, datos de entrada, operador de implicación y método desborrosificador). Este problema que también es planteado en [20], consiste en diseñar un controlador de una máquina lavadora inteligente que calcule el tiempo óptimo de lavado a partir de una serie de parámetros, grado de suciedad (Dirtiness) y el tipo de suciedad (Type of dirt), obtenidos mediante sensores ópticos según el esquema de la figura N.º 6. Los resultados indicados en la referencia [19] se tomarán como base para contrastarlos con los obtenidos mediante el sistema FISI Logic.

El esquema general del problema se muestra en la Figura N.º 6:

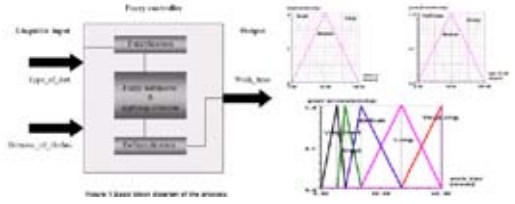


Figura N.º 6.

La solución mediante FISI Logic inicia creando un proyecto.

Figura N.º 7.

Se crearon las variables de entrada (Dirtness y Type_of_dirtness) y salida (Wash_Time) mediante el primer módulo. Estas son presentadas en la interfaz principal mediante un árbol de contenido.

Figura N.º 8. Especificación de los datos de la variable de salida WASH_TIME.

Se editó la *base de reglas borrosas* según las reglas definidas en [12].

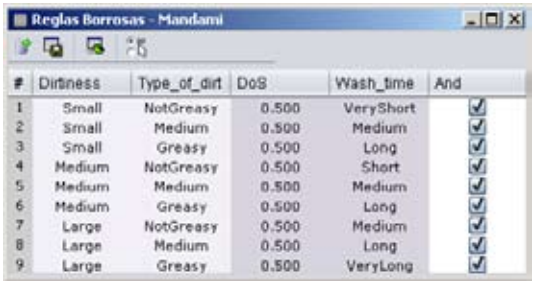


Figura N.º 9. BRB.

Mediante la opción de menú *Ingreso de Datos* enseguida se borrosificaron los valores de las variables de entrada (datos tomados de [19]); los cuales son 30 y 70 para DIRTINESS y TYPE_OF_DIRT, respectivamente.

Figura N.º 10. Borrosificador.

El Mecanismo de Inferencia permite elegir el operador de implicación, que muestra las reglas disparadas y el resultado final de la inferencia. Se pudo comprobar que coinciden con la solución del ejemplo original: los términos lingüísticos de la variable de salida (Medium) con valor de pertenencia 0.6 y (Long) con valor 0.4.

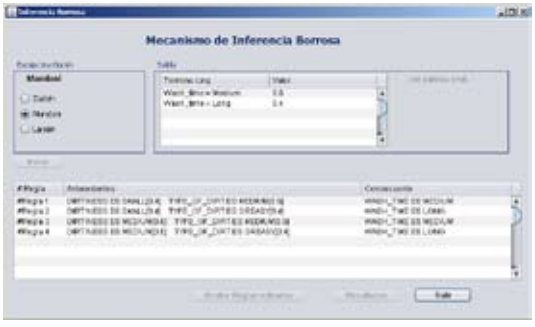


Figura N.º 11. Mecanismo de inferencia.

El módulo de desborrosificación es activado de forma instantánea, mostrando los conjuntos borrosos resultantes según el paso anterior (Medium y Long). Se elige enseguida el método de desborrosificación Suma de Centro de Areas (SCoA) cuyos fundamentos son muy parecidos a los del Centro de Areas (CoA) usado en el problema [19].

Figura N.º 12. Desborrosificador.

Finalmente el sistema muestra el resultado para el problema siendo el tiempo de lavado, WASH_TIME, aproximadamente 26.76 minutos). Este valor está bastante próximo del indicado en la gráfica tridimensional de la Figura 13 (tomada de [19]), donde la flecha por debajo del valor 30 indica la respuesta el sistema.

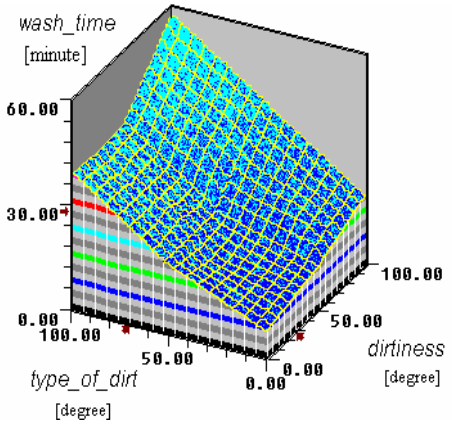


Figura N.º 13. Salida tridimensional del sistema

Análisis de resultados

6. Se puede afirmar que el módulo de creación de las variables del sistema cumple su objetivo. La creación de las variables se realiza de forma intuitiva y la previsualización de las funciones de pertenencia y sus parámetros son de gran ayuda para que el

usuario pueda definir adecuadamente los términos lingüísticos de cada variable.

7. El editor de reglas borrosas es una herramienta que permite que el usuario registre, de un modo sencillo y mediante una interfaz amigable, las reglas asociadas al ámbito del conocimiento del problema a tratar.
8. Se realiza de forma eficiente el ingreso y edición de la función de salida perteneciente a las reglas borrosas de los sistemas Sugeno.
9. Es posible aplicar los operadores de implicación considerados, verbigracia los de Zadeh, Mamdani y Larsen; aunque el de Zadeh no se usa para aplicaciones de ingeniería se ha considerado para el mecanismo de inferencia dicho operador. Este módulo muestra de forma detallada el total de reglas con sus grados de membresía, las reglas disparadas y los valores de salida
10. Obtiene también el mayor valor cuando hay dos o más reglas que generan el mismo consecuente con diferentes grados de pertenencia. Asimismo, une los conjuntos borrosos resultantes de la inferencia.
11. Se puede afirmar, a partir de lo indicado en los dos puntos inmediatamente anteriores, que el mecanismo de inferencia implementa correctamente sus características básicas de diseño.
12. El módulo de desborrosificación obtiene con buena aproximación el valor de salida del SIB, al comparársele con el reportado en [19]. La diferencia entre ambos valores se debe a que FISI Logic usa el método de desborrosificación llamado Suma de Centros de Áreas y en la referencia se reporta haber usado el Centro de Áreas. Si se comparasen estos resultados con el obtenido por cálculo manual también habría una diferencia puesto que la etapa de desborrosificación se hace por integración analítica y no numérica.
13. El módulo de desborrosificación no se conecta con la base de datos para obtener los puntos con los cuales calculará la integración numérica del conjunto borroso de salida, el cual es pasado por el módulo de inferencia. Para efectuar ese cálculo, los toma a partir de una lista en ejecución.
14. La fundamentación teórica de los sistemas basados en lógica borrosa requerida para desarrollar el proyecto se basó en las clases impartidas en el curso Sistemas Inteligentes, y fue complementada por varias de las referencias enumeradas en este artículo; de especial importancia para el desarrollo de FISI Logic es la referencia [10] en la que se presenta una síntesis de esta teoría y se plasma el conocimiento previo al proyecto.
15. La metodología Programación Extrema se mostró útil para el desarrollo del proyecto en el plazo de tiempo programado.
16. Los resultados obtenidos con FISI Logic muestran que sus módulos y el sistema en general, cumplen sus características básicas de diseño. Estos resultados y la validación de los mismos, alientan positivamente su uso para modelar y resolver problemas de lógica borrosa.
17. Asimismo dichos resultados califican a FISI Logic como una herramienta didáctica para el aprendizaje y entendimiento de los sistemas de inferencia borroso.

Dificultades:

18. El modelamiento inicial de la base de datos para representar los sistemas de inferencia borrosa conllevó tiempo para la investigación y abstracción del mismo.
19. La conexión con la base de datos no se pudo concretar en los módulos de mecanismo de inferencia y desborrosificación por motivos de tiempo y de priorizar funcionalidades.
20. Los módulos de creación de variables y desborrosificación presentaron un alto nivel de dificultad en su programación.
21. El uso de las librerías especializadas JFreeChart y Flanagan mencionadas en [17] y [18] tuvo un costo considerable de aprendizaje para su uso en la codificación.
22. La estandarización de las interfaces tomó dos versiones de prueba.
23. La integración de los módulos con la interfaz de usuario fue una de las etapas más complicadas en el desarrollo del proyecto

Limitaciones:

24. No está dentro del alcance de la herramienta computacional desarrollada la inferencia de variables con universo de discurso discreto.
25. No se muestra correctamente la pertenencia de los conjuntos borrosos activados en la inferencia, ni se

muestra la unión de los conjuntos borrosos resultantes de la inferencia Mamdani.

26. No se guarda la configuración del sistema de inferencia borrosa.

CONCLUSIONES

1. Se alcanzó la meta planteada de construir en el plazo predeterminado, verbigracia, la duración de un semestre, una nueva herramienta para la implementación de sistemas de inferencia borroso con componentes básicos. Este sistema computacional desarrollado grupalmente permite en su primera versión, con las limitaciones indicadas, diversas aplicaciones tales como sistemas de apoyo en la toma de decisiones, control de máquinas, etc.
2. Este objetivo no se hubiese podido cumplir sin la participación eficiente de los equipos de trabajo conformados por los alumnos del curso, que con el esfuerzo desplegado y su capacidad en programación orientada a objetos, consiguieron construir el sistema software, a partir de la aplicación de librerías especializadas y de los conocimientos adquiridos en clase sobre esta técnica inteligente.
3. Una de las mayores dificultades fue el coordinar el trabajo de todos los grupos puesto que el proceso de integración del sistema debió realizarse continuamente a lo largo del ciclo conforme se producía el avance de los grupos en los respectivos módulos.
4. La construcción del sistema FISI Logic mostró una vez más que la aplicación de metodologías de desarrollo es importante en el desarrollo de sistemas computacionales de cualquier área, en el presente caso la inteligencia artificial, y no solo para sistemas de información empresarial. En futuras versiones se podrá usar otras metodologías de desarrollo, especialmente las empleadas para los sistemas basados en conocimiento tal como la denominada CommonKADS.
5. La herramienta construida constituye por sí misma un valioso medio para el aprendizaje de los sistemas basados en lógica borrosa debido a que facilita la presentación de los ejemplos prácticos o casos de estudio en los que se aplique esta técnica. La participación de los alumnos del curso Sistemas Inteligentes en las fases de diseño y desarrollo de un sistema software como el desarrollado, debería redundar en un mejor conocimiento práctico de estos sistemas, complementando la teoría ofrecida en clases sobre este tema.
6. La idea de construir una herramienta de propósito general para sistemas basados en lógica borrosa planteada y ejecutada de forma conjunta con los alumnos de un determinado curso durante un semestre, se puede generalizar para otros sistemas inteligentes, y quizás para otro tipo de sistemas computacionales.
7. El sistema software desarrollado queda como plataforma base para futuros desarrollos en los que, además de resolver sus limitaciones, será posible ampliar sus características y funcionalidades de modo que pueda constituirse en una herramienta de mayor capacidad para la implementación de sistemas de inferencia difusa.
8. Finalmente, el proyecto desarrollado demuestra que es posible realizar investigación en la Facultad de Ingenieria de Sistemas, aunque sea a pequeña escala.

AGRADECIMIENTOS

El coautor del presente trabajo y profesor del curso Sistemas Inteligentes de nuestra facultad, desea expresar su agradecimiento a los alumnos de dicho curso en el semestre 2009-2 por el esfuerzo desplegado para la concreción del objetivo planteado, con especial mención a los alumnos Marcos Sobrevilla Cabezudo, Renzo Araos Chea, Dayhana Matos y John Quispe Ochoa quienes colaboraron decididamente para el buen término del mismo.

REFERENCIAS BIBLIOGRÁFICAS

[1] Lofti Zadeh. Fuzzy Sets. Information&Control, 8, 338-353. 1965.

[2] Fuzzytech. En http://www.fuzzytech.com/

[3] Fuzzy Logic Toolbox. http://www.mathworks.com/products/fuzzylogic/

[4] jFuzzyLogic. http://jfuzzylogic.sourceforge.net/html/index.html

[5] José Juárez et al. Control (capítulo de muestra). En: http://novella.mhhe.com/sites/dl/free/8448156188/592196/8448156188_CapMuestra.pdf. Accesado en Febrero 2010.

[6] Fuzzyclips. http://www.ortech-engr.com/fuzzy/fzyclips.html

[7] FIDE. http://www.aptronix.com/fide/fide.htm

[8] I. Baturone et al. Using Xfuzzy Environment for the Whole Design of Fuzzy Systems. Proc. IEEE International Conference on Fuzzy Systems. En: http://www.imse-cnm.csic.es/online/2007/FUZZIEEE2007.IBC.pdf (Accesado en Febrero 2010)

[9] George Klir y Bo Yuan. FUZZY SETS AND FUZZY LOGIC. Theory and Applications. Ed. Prentice Hall. 1995

[10] Rolando Maguiña Pérez. Sistemas de inferencia basados en Lógica Borrosa: Fundamentos y Caso de estudio. Revista RISI. 2010 (por publicar)

[11] B. Martín del Brío, A. Sanz Molina. Redes Neuronales y Sistemas Difusos. Ed. Alfaomega. 2000.

[12] SELECTING A DEVELOPMENT APPROACH. En: http://www.cms.hhs.gov/SystemLifecycleFramework/Downloads/SelectingDevelopmentApproach.pdf (Accesado en Febrero 2010)

[13] Metodología de desarrollo de software. En: http://es.wikipedia.org/wiki/Metodolog%C3%ADa_de_desarrollo_de_software (Accesado en Febrero 2010)

[14] Paco Blanco et al.. Metodología de desarrollo ágil para sistemas móviles Introducción al desarrollo con Android y el iPhone. Doctorado en Ingenieríad e Sistemas Telemáticos, Universidad Politécnica de Madrid. 2009. http://lab.gsi.dit.upm.es/apache/adam/homepage/wp-content/files/docsCursos/Agile_doc_TemasAnv.pdf

[15] Programación Extrema. Portal VerSoft. http://www.verasoft.es/web/guest/xp (Accesado en Febrero 2010)

[16] Miguel Jaque Barbero. Cuaderno de notas. Programación extrema. http://migueljaque.com/index.php/metodologias/xp (Accesado en Febrero 2010)

[17] JFreeChart. http://www.jfree.org/jfreechart/ (Accesado en Diciembre 2009)

[18] Michael Thomas Flanagan's Java Scientific Library. http://www.ee.ucl.ac.uk/~mflanaga/java/ (Accesado en Diciembre 2009)

[19] Manish Agarwal. Fuzzy Logic Control of Washing Machines. Indian Institute of Technology, Kharagpur. http://softcomputing.tripod.com/sample_termpaper.pdf (Accesado en Enero 2010)

[20] Site de empresa Aptronix. En: http://www.aptronix.com/fuzzynet/applnote/wash.html (Accesado en Febrero 2010)