UD-SHELL: UNA HERRAMIENTA SOFTWARE PARA LA CONSTRUCCIÓN DE BASES DE CONOCIMIENTO

Tecn. NAIDA MILENA CHITIVA*
Ing. JORGE ENRIQUE RODRÍGUEZ RODRÍGUEZ**
jrodri@atlas.udistrital.edu.co

1. Introducción

A través del tiempo se han venido desarrollando Sistemas Expertos (SE) y Sistemas Basados en Conocimiento (SBC)[1] que ejecutan una amplia y complicada variedad de tareas que en el pasado solamente podían llevarse a cabo por un número limitado de expertos humanos. Así mismo se han implementado herramientas llamadas *shell* o *motor de inferencia*[2], las cuales cumplen una función similar a los SE y SBC. La diferencia consiste en que los *shell* no contienen conocimiento, lo cual resulta ser una ventaja dado que con ellos se pueden crear bases de conocimiento en cualquier área, permitiendo el desarrollo de múltiples SE y SBC. Estas herramientas de software se componen de un motor de inferencia y de utilidades que además de la constitución de bases de conocimiento permiten otras funciones relacionadas con los usuarios que se aplican en diferentes dominios y situaciones.

El desarrollo masivo de los *shells* (Guru, Microexp, UC-Shell y Mycin, entre otros) se ha dado en países industrializados desde hace aproximadamente tres décadas; en contraste, en nuestro país se identifican muy pocas experiencias similares, a tal punto que solamente se conocen dos *shells:* uno desarrollado por la Univer-

PALABRAS CLAVES

MOTOR DE INFERENCIA, SISTEMAS BASADOS EN EL CONOCIMIENTO, REPRESENTACIÓN DEL CONOCIMIENTO, RAZONAMIENTO ADELANTE, RAZONAMIENTO ATRÁS, LENGUAJE UNIFICADO DE MODELADO Y TÉCNICAS ORIENTADAS POR OBJETOS

* Tecnóloga en Sistematización de Datos Universidad Distrital F.J.C.
** Ingeniero de Sistemas, Especialista en Telemática, Especialista en Ingeniería del Software y estudiante de la Maestría en Ingeniería de Sistemas de la Universidad Nacional de Colombia. Profesor de tiempo completo adscrito a la Facultad Tecnológica de la Universidad Distrital F.J.C.

1 Sistemas informáticos que buscan simular los procesos de memorización, acción, aprendizaje y comunicación de un experto humano en determinada área del conocimiento; con el fin de ofrecer una herramienta para el apoyo de la toma de decisiones Los SE y SBC se han aplicado casi a todos los campos del conocimiento; algunos se han diseñado como herramientas de investigación, mientras que otros satisfacen importantes funciones de negocios e industrias.

2 Contiene los algoritmos para deducir conclusiones o soluciones para el usuario, basado en el conocimiento plasmado en la base de conocimiento

sidad Industrial de Santander (Esbac), y el otro por la Universidad Nacional de Colombia (UN-Shell).

Con *UD Shell* se pretende ofrecer a los desarrolladores de SE y SBC una herramienta de software alternativa a las existentes. Ella se compone básicamente de un motor de inferencia que permite trabajar diferentes dominios asociados con sus métodos de inferencia. Así mismo se busca incentivar a estudiantes y profesionales de diferentes áreas del conocimiento para que investiguen y masifiquen el desarrollo de SE y SBC, las cuales se constituyen en importantes aplicaciones de la Inteligencia Artificial (IA)[3].

2. Modelos Funcionales de SE

En la siguiente tabla se muestran los modelos funcionales de los SE, el tipo de problema que intentan resolver y algunos de sus usos específicos.

Dentro de los elementos más importantes de la arquitectura de un SBC/ SE se encuentran:

- **Representación del conocimiento.** Considerando que el conocimiento es importante y primordial para el comportamiento inteligente, su representación constituye una de las máximas prioridades de investigación en IA. En organismos biológicos se estima que él es almacenado en forma de estructuras complejas de neuronas interconectadas; en los computadores también se almacena como estructuras simbólicas, pero en forma de estados eléctricos y magnéticos. En forma natural, los seres humanos representan el conocimiento simbólicamente: imágenes, lenguaje hablado y escrito; sin embargo también se han desarrollado otros sistemas de representación: literal, numérico, estadístico y lógico. Así, la

CATEGORIA	TIPO DE PROBLEMA	USO
Interpretación	Deducir situaciones a partir de datos observados	Análisis de imágenes, reconocimiento del habla, inversiones financieras
Predicción	Inferir posibles consecuencias a partir de una situación	Predicción metereológica, previsión del tráfico, evolución de la bolsa
Diagnóstico	Deducir fallos a partir de sus efectos	Diagnóstico médico, detección de fallos en electrónica
Diseño	Configurar objetos bajo ciertas especificaciones	Diseño de circuitos, automóviles, edificios, etc.
Planificación	Desarrollar planes para llegar a unas metas	Programación de proyectos e inversiones. Planificación militar
Monitorización o supervisión	Controlar situaciones donde hay planes vulnerables	Control de centrales nucleares y factorías químicas
Depuración	Prescribir remedios para funcionamientos erróneos	Desarrollo de software y circuitos electrónicos
Reparación	Efectuar lo necesario para hacer una corrección	Reparar sistemas informáticos, automóviles, etc.
Instrucción	Diagnóstico, depuración y corrección de una conducta	Corrección de errores, enseñanza
Control	Mantener un sistema por un camino previamente trazado. Interpreta, predice y supervisa su conducta	Estrategia militar, control de tráfico aéreo
Enseñanza	Recoger el conocimiento y mostrarlo	Aprendizaje de experiencia

Tabla 1. Modelos Funcionales de los Sistemas Expertos

[3] La IA estudia cómo hacer que las máquinas hagan cosas que hasta el momento sólo hacían bien los seres humanos

ingeniería cognoscitiva[4] ha adaptado diversos sistemas de representación, que implantados en un computador se aproximan mucho a los modelos elaborados por la *psicología cognoscitiva*[5] para el cerebro humano. Entre los principales se tienen:

- **Lógica Simbólica Formal:** lógica proposicional, lógica de predicados y sistemas de producción
- **Formas Estructuradas:** redes asociativas, estructuras marco, representación orientada a objetos.

Los *sistemas de producción* proporcionan una estructura que facilita la descripción y ejecución de un proceso de búsqueda. Consisten en un conjunto de facilidades para la definición de reglas, mecanismos para acceder a una o más bases de conocimientos y datos, una estrategia de control que especifica el orden en el que las reglas son procesadas, la forma de resolver los conflictos que pueden aparecer cuando varias reglas coinciden simultáneamente y un mecanismo que se encarga de aplicar las reglas.

Las reglas son un importante paradigma de representación del conocimiento. Los sistemas basados en ellas son entonces los utilizados más comúnmente. Su simplicidad y similitud con el razonamiento humano han contribuido a su popularidad en diferentes dominios.

Las reglas representan el conocimiento utilizando un formato ***Si - Entonces*** (***If-Then***), es decir tienen dos partes: la parte ***Si*** (***If***) es el antecedente, premisa, condición o situación; la parte ***Entonces*** (***Then***) es el consecuente, conclusión, acción o respuesta. Ellas pueden ser utilizadas para expresar un amplio rango de asociaciones, por ejemplo:

SI está manejando un vehículo Y se aproxima una ambulancia, ENTONCES baje la velocidad Y hágase a un lado para permitir el paso de la ambulancia

SI su temperatura corporal es de 39° C, ENTONCES tiene fiebre.

SI el drenaje del lavabo está tapado Y la llave de agua está abierta, ENTONCES se puede inundar el piso.

- **Proceso de razonamiento.** Es una progresión desde un conjunto inicial de afirmaciones y reglas hacia una solución, respuesta o conclusión; sin embargo, vale destacar que los resultados obtenidos pueden variar significativamente, pues a partir de los datos conocidos se va avanzando hacia la solución; el proceso es denominado *guiado por los datos*, o de encadenamiento progresivo o hacia adelante (*forward chainning*). También se puede seleccionar una posible solución y tratar de probar su validez buscando evidencia que la apoye. Este proceso se denomina *guiado por el objetivo* o de encadenamiento regresivo o hacia atrás (*backward chainning*).

3. ¿Por qué UD Shell?

Las herramientas de software existentes que han sido desarrolladas con este tipo de caracterísi-

[4] La ingeniería cognoscitiva involucra la representación del conocimiento heurístico amplio, impreciso, mal definido, que está almacenado en la mente de pocos expertos. Debido a que el conocimiento heurístico no es muy conocido ni entendido, para lograr transferirlo desde las mentes de los individuos que lo poseen hasta una representación computarizada, tienen que ser utilizadas ciertas técnica. La transferencia se denomina "adquisición de conocimiento", es elaborada y consume mucho tiempo

[5] Rama de la psicología que se ocupa de los procesos a través de los cuales el individuo obtiene conocimiento del mundo y toma conciencia de su entorno, así como de sus resultados.

cas ofrecen un único método de inferencia. Microexpert y Prolog, por ejemplo, utilizan encadenamiento hacia atrás; por su parte, Clips y OPS5 utilizan encadenamiento hacia delante. La herramienta alternativa que aquí se presenta es fundamentalmente un motor de inferencia que permita trabajar mediante encadenamiento hacia adelante y hacia atrás.

El propósito es masificar su uso especialmente entre estudiantes, considerando que generalmente estos sistemas son construidos por ingenieros de conocimiento y/o personas con gran experiencia en esta área. El actual desconocimiento o imposibilidad de acceso a los *shells* conlleva a un desinterés en esta área de la IA por la mayor parte de las personas, pero en especial por estudiantes. UD Shell podrá adquirirse fácilmente, a diferencia de la dificultad de acceder a las herramientas extranjeras.

4. ¿Cómo se desarrolló UD Shell?

Para el desarrollo de esta herramienta se optó por utilizar una metodología orientada a objetos y un lenguaje de modelado (UML[6]), que permite la descripción de los modelos lógicos y físicos poseídos por UD Shell en el proceso de creación y ejecución de Bases de Conocimiento. Además, la notación de UML ofrece una mejor comprensión, ya que ella es muy conocida en el ámbito de los desarrolladores de software.

Para el análisis de requisitos se utilizó un Diagrama de Casos de Uso (ver Figura 1), el cual permite representar las acciones que realiza UD Shell, junto con los diferentes actores que interactúan.

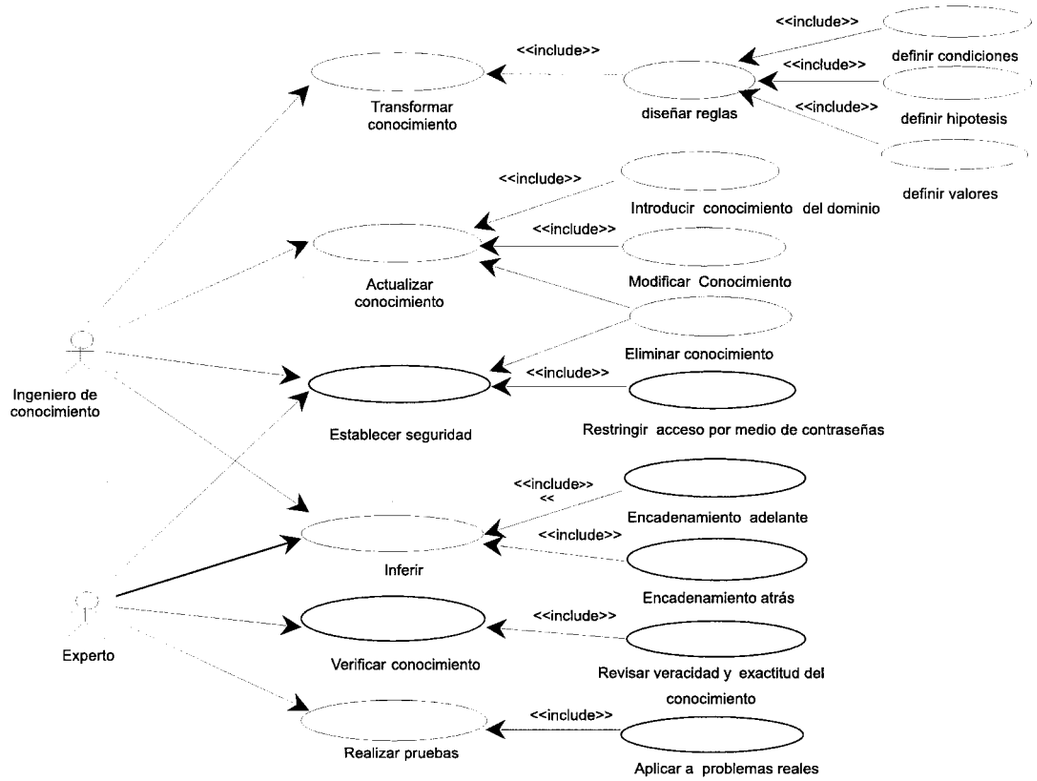

Figura 1. Diagrama de Casos de Uso para UD Shell

[6] Lenguaje unificado de modelado

La codificación de la herramienta se realizó en Java[7], debido a su independencia de la plataforma[8]. Este código binario de salida es un conjunto de instrucciones altamente optimizado, diseñado para ser ejecutado por una máquina virtual que emula el intérprete de Java. Debido a que los programas Java son interpretados en lugar de compilados es mucho más fácil ejecutarlos en gran variedad de entornos; una vez que existe el programa de ejecución para un sistema dado, cualquier programa Java puede ejecutarse en esa plataforma. Los archivos no ocupan mucho espacio en disco; esto hace de este lenguaje una herramienta portátil y compatible.

5. ¿Cómo funciona UD Shell?

UD Shell permite desarrollar Bases de Conocimiento por medio de sistemas de producción, con el empleo de dos métodos de inferencia: encadenamiento hacia delante y hacia atrás. La herramienta posibilita el manejo de archivos para el almacenamiento en disco de las Bases de Conocimiento; además cuenta con un buen nivel de seguridad para el acceso a ellas por medio de contraseñas, las cuales también restringen o amplían las posibilidades de manejo, y con métodos de explicación para justificar cómo y por qué se llega a cierta conclusión, partiendo del dominio de la Base de Conocimiento y de la Base de Hechos. El cómo se comprueba visualizando las reglas ejecutadas; para saber el por qué se visualiza la descripción de hipótesis y condiciones que el experto facilitó en la creación de las reglas.

Las principales funciones de UD Shell son:

- Crear Bases de Conocimiento (reglas, hipótesis y condiciones)
- Cargar Bases de Conocimiento
- Guardar Bases de Conocimiento
- Actualizar los hechos de las Bases de Conocimiento
- Realizar inferencia por medio de 2 métodos (encadenamiento adelante y atrás)
- Visualizar reglas
- Imprimir reglas
- Cambiar clave a las Bases de Conocimiento.

El entorno de UD Shell consta de una ventana con una barra de menús y sus respectivos submenús, una barra de herramientas, una ficha con 2 páginas y una barra de estado. Al abrir una Base de Conocimiento el usuario tiene dos opciones; los modos de acceso son definidos por los tipos de usuario, que pueden ser "*experto*" e "*Ingeniero del Conocimiento*".

A través del tipo de usuario *experto* puede trabajarse sólo en modo de lectura, es decir, puede realizarse visualización de reglas e inferencia a partir de ellas mediante encadenamiento hacia adelante y hacia atrás, sin poder realizar ningún tipo de modificación. Al acceder mediante la opción *Ingeniero de Conocimiento* se adquiere el manejo completo de la aplicación, sin restricción alguna. En los dos casos se solicitan contraseñas de acceso, con posibilidad de redigitación en caso de equivocación.

Figura 2. Entorno Gráfico de UD Shell

[7] Java es un lenguaje de programación orientado a objetos (OO) de propósito general desarrollado por Sun Microsystems a principios de 1991. La idea de Sun es apuntarlo en Internet y saturar el mercado de CPU´s que interpreten este lenguaje, con una conexión a la red, un monitor, un teclado y un ratón. Es lo que ya se han llamado el Network Computer (NC), que Sun espera se llame el Java Computer (JC)

[8] La salida de un compilador Java, no es código ejecutable sino que es código binario (bytecode)

En UD Shell el conocimiento se representa tomando como base los sistemas de producción, ya que estos proporcionan una estructura que facilita la descripción y ejecución de los procesos de búsqueda y representación: las *reglas de producción.* Su simplicidad y similitud con el razonamiento humano han contribuido a su popularidad en diferentes dominios.

El proceso de actualización en UD Shell consiste en el ingreso, modificación o eliminación de reglas, hipótesis y/o condiciones. El proceso puede realizarse en el área de trabajo ubicada en la página denominada *Actualizar*, localizada en la ficha de la ventana principal de la aplicación. Para el control de los datos en las Bases de Conocimiento solamente se permite el manejo de los siguientes caracteres (letras y dígitos):

Figura 3. Modo de Acceso

- Caracteres correspondientes al alfabeto, incluyendo la "ñ"
- El carácter subguión (_), para los casos en los cuales hay más de una palabra
- Los operadores señalados en la actualización de condiciones (=, !=, <, <=, >, >=)
- En la parte de descripción si se permiten los demás caracteres, en caso de que el usuario quiera agregar de una manera especial las explicaciones.

UD Shell permite el manejo total del teclado (botones, menús y listas que aparecen su entorno). El desplazamiento hacia delante por los componentes mediante el teclado se realiza mediante la tecla *Tab* o *Ctrl+Tab;* para desplazarse hacia atrás se debe utilizar *Shift + Tab*.

Para proceder al diseño de una Base de Conocimiento primero es necesario estructurar el conocimiento del problema a resolver en forma de reglas, con sus respectivas hipótesis y conclusiones; ellas permiten ingresar el conocimiento a la nueva base. El siguiente es un ejemplo de descripción de las características de un ser humano en forma de reglas:

1. CLASE = MAMÍFERO	**2. ORDEN = PRIMATE**	**3. ANIMAL = HOMBRE**
tipo = vertebrado	clase = mamífero	orden = primate
reproducción = vivípara	dedos = cinco	cráneo = semiredondo
locomoción = patas	dedos _ con = uñas	visión = binocular
respiración = pulmonar	patas = cortas	postura = erecta
	colmillos = cortos	piel = desnuda
		razona = si

Tabla 2. Ejemplo de una Base de Conocimiento

Como se ha dicho, en el entorno de *ingeniero de conocimiento* se pueden modificar las reglas inicialmente planteadas; sin embargo, los autores del artículo omiten los detalles para su actualización[9].

Figura 4. Visor de Reglas

[9] Para mayor información se hace necesario remitirse al manual de usuario de UD Shell

El proceso de inferencia consiste en la utilización del dominio almacenado en la Base de Conocimiento para generar nuevo conocimiento. Así, con la información que se encuentra en la Base de Conocimiento y la que se solicita al usuario puede iniciarse un proceso de búsqueda de soluciones empleando uno de los dos métodos de inferencia disponibles ya citados. Es claro que si el usuario no brinda la información necesaria para el proceso de inferencia no podrán encontrarse soluciones.

Figura 5. Entorno de Inferencia UD Shell

El proceso de inferencia se realiza en la segunda página de la ficha de la ventana principal de UD Shell. El orden en que las reglas son procesadas es el mismo en el cual aparecen en la Base de Conocimiento. Además, cuando el atributo y el valor contienen dígitos en una condición el sistema se encarga de verificarla automáticamente sin necesidad de consultar al usuario, pues tiene la capacidad de realizar las operaciones matemáticas correspondientes. Si el atributo contiene símbolos (cadenas de caracteres) y el valor dígitos, la aplicación solicita el valor del símbolo contenido en la parte atributo.

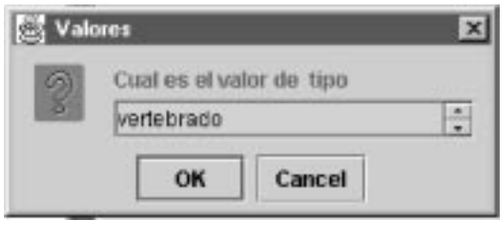

Figura 6. Solicitud de Información

La siguiente es la explicación de la lógica y forma de uso de cada uno de los métodos de inferencia. Si no existe una selección previa de método, el panel de inferencia aparece inicialmente vacío.

- **Encadenamiento hacia adelante.** El proceso parte de la información que el usuario suministra, la cual se introduce en la base de hechos; a partir de ella UD Shell comienza a deducir nuevos hechos que se pueden agregar a esta base, teniendo en cuenta que una regla se puede disparar si todas sus condiciones son ciertas. Si hay reglas que puedan dispararse, su conclusión se agrega a la base de hechos como un nuevo hecho.

Para ejecutar el proceso, primero se selecciona la opción *encadenamiento hacia adelante* del menú Inferir; el panel correspondiente (ver Figura 8) contiene inicialmente una lista correspondiente a la Base de Hechos y un botón llamado *Hechos*, encargado de iniciar el proceso de inferencia bajo este método haciendo *click* u oprimiendo *enter* en él. Luego aparece un diálogo solicitando información al usuario de acuerdo con la Base de Conocimiento activa; si el valor del atributo evaluado es dígito se muestra un cuadro de entrada de datos, y UD Shell seguirá solicitando información hasta que se crea conveniente, de acuerdo con las reglas que existan. En la lista de opciones hay una denominada *"Ignorado"*, que el usuario podrá escoger si ignora el valor del atributo que en ese momento se solicita. Toda la información se va almacenando en la Base de Hechos.

Entre las herramientas se dispone de una barra de progreso que permite a los usuarios tener idea del estado de avance de la inferencia; en caso de desearse interrumpir el proceso antes de que este termine puede seleccionarse *Cancelar* en el cuadro de solicitud de Información; en tal caso el sistema informará que el proceso de inferencia quedó incompleto. Después de que el sistema termina la solicitud de información empieza a revisar las reglas que puede ejecutar o disparar, de acuerdo con los hechos suministrados, procede a la ejecución y a agregar las conclusiones a la base de hechos. Cuan-

Figura 8. Inferencia Encadenamiento Hacia Adelante

do el proceso termina se da un mensaje de *Inferencia Terminada*, y en la lista de "*reglas ejecutadas*" se agregan el número de regla y su hipótesis para dar a conocer al usuario aquellas que pudieron ejecutarse de acuerdo con la información incorporada en la base de hechos. Si se quiere observar por qué se ejecutaron las reglas del listado se hace *click* sobre la regla que se quiere analizar; luego de ello aparece un diálogo que muestra toda la regla, para que el usuario puede comparar sus condiciones de regla con los hechos suministrados en la Base de Hechos. Existe también un botón denominado *"Nueva inferencia"*, para permitir al usuario iniciar un nuevo proceso, si así lo desea.

Para el ejemplo planteado, al realizar la inferencia hacia delante se puede observar que los hechos solicitados por UD Shell se han introducido como verdaderos de acuerdo con las reglas existentes, es decir, se les ha dado el mismo valor con el cual se encuentran en la Base de Conocimiento para poder ejecutar las tres reglas existentes.

Si en el momento de creación de la Base de Conocimiento se digitó alguna explicación en los campos de texto *descripción de hipótesis* y *condiciones*, después de visualizar el cuadro de diálogo anterior puede observarse otro que explica el por qué de la regla, en el cual se mostrará la explicación que se almacenó en los campos de textos mencionados.

- **Encadenamiento hacia atrás.** En este caso el proceso parte de los objetivos, es decir, de una hipótesis, intentándose verificar si se satisface o no. UD Shell examina las reglas que se pueden disparar para concluir que la hipótesis es cierta; si ellas no se pueden ejecutar con la información disponible se tienen que evaluar, convirtiéndose en un subobjetivo para ser verificado. Por lo tanto es este un proceso recursivo que tiene fin sólo cuando se comprueba el objetivo (hipótesis inicialmente sugerida) o cuando no existen más reglas para disparar. Este método de inferencia tiene una gran ventaja: solicita únicamente la información necesaria para ejecutar las reglas relacionadas con la hipótesis sugerida.

En UD Shell este método de inferencia inicia visualizando el pánel de encadenamiento hacia atrás, el cual contiene una lista con todas las hipótesis de las reglas existentes, y un botón llamado "Inferir", encargado de iniciar el proceso. De esta forma, primero se tiene que escoger la hipótesis a comprobar de la lista correspondiente para iniciar el proceso; en caso de no escogencia previa se muestra un mensaje advirtiendo de la falta.

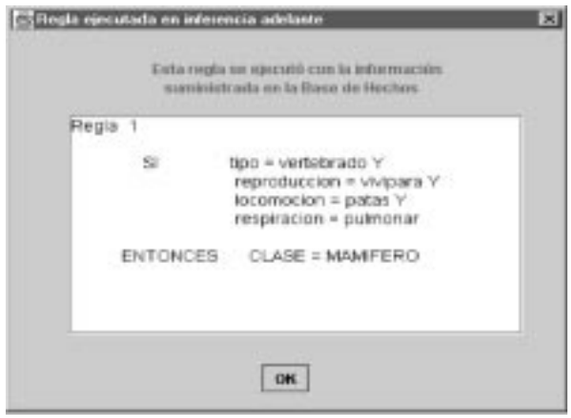

Figura 9. Diálogo Regla Ejecutada

Para el ejemplo se escogió la hipótesis de la Regla # 3 para ser comprobada, y también se dio un valor verdadero a los hechos solicitados por la Base de Conocimiento. UD Shell revisa las reglas ejecutables para realizar la verificación requerida, en caso que existan las evalúa y encadena hasta agotar todas las reglas posibles o hasta que llegue a una que no dependa de otras (compuesta de hechos simples y no de conclusiones de otras). Cuando llega a estos hechos solicita al usuario el valor de cada uno de ellos; si las respuestas satisfacen las reglas del subobjetivo de la hipótesis inicial se irán ejecutando las reglas hasta llegar a aquella de la cual se sugirió la hipótesis; así se habrá realizado la verificación requerida.

La solicitud de información se realiza de manera similar a la presentada en la Figura 9 y los valores suministrados se van almacenando en la Base de Hechos. Después de realizar la inferencia con los hechos incorporados en la Base de Hechos se ejecutan o no las reglas necesarias, y el número de éstas, junto con sus hipótesis, se almacenan en la lista "Reglas Ejecutadas". Después se muestra un diálogo diciendo si el objetivo es verdadero o no, es decir, si la hipótesis sugerida se comprobó o no.

Figura 10. Selección de Hipótesis Inferencia hacia Atrás

Entonces el usuario podrá ver las reglas que fue necesario ejecutar para poder comprobar el objetivo.

Para observar una o varias de las reglas que se ejecutaron, el usuario debe seleccionar en la lista "Reglas Ejecutadas" la que desea ver y, si lo desea, compararla con los hechos incorporados en la Base de Hechos. Estas reglas se muestran en un diálogo similar a la Figura 9; la diferencia es el título del diálogo. Al igual que en el Encadenamiento hacia delante, si hay una descripción de hipótesis y condiciones en los campos de texto de los mismos, se podrá visualizar esta explicación al igual que se visualizan las reglas. Este pánel también contiene una barra de progreso que le permite al usuario ver el avance de la inferencia.

En el ejemplo que se realizó, se dio a conocer la manera como se debe realizar la inferencia con cualquiera de los dos métodos, para cualquier Base de Conocimiento, sin importar su número de reglas; para la inferencia con encadenamiento hacia delante entre más reglas hayan, más datos se solicitarán al usuario y el proceso será un poco más largo, pero en cuanto a lo demás, el procedimiento es igual al que se acabo de realizar.

Esta es la mayor descripción posible de la inferencia de UD Shell; para adquirir mayor conocimiento sobre ella se debe practicar creando e infiriendo con nuevas Bases de Conocimiento y también tomando como ejemplo las que se suministran junto con este trabajo.

El cuadro de diálogo "Acerca de UD Shell" que se puede abrir desde el menú ayuda, contiene todos los datos relacionados con el desarrollo de esta herramienta.

Conclusiones

- El desarrollo de UD Shell permite masificar la construcción de SE y SBC por aquellas personas que tengan interés en el aprendizaje y aplicación de técnicas de Inteligencia Artificial

  UD Shell suministra dos métodos de inferencia para la búsqueda de nuevo conocimiento: el encadenamiento hacia delante y el encadenamiento hacia atrás, característica que no poseen los motores de inferencia que están en el mercado
- Como forma de representación del conocimiento UD Shell utiliza los sistemas de pro-

ducción, partiendo del hecho que ésta es una de las formas más utilizadas en la implementación de SE y SBC

- Con este motor de inferencia se ofrece una herramienta de fácil acceso a todas aquellas personas interesadas en la construcción de SBC y SE, ya que en nuestro mercado este tipo de herramientas no es muy asequible
- El conocimiento en UD Shell se almacena de manera similar a la humana, lo que permite que cualquier persona comprenda el dominio incorporado en la Base de Conocimiento.
- Para el desarrollo de UD Shell se utilizó una metodología de desarrollo, un lenguaje de modelado y un lenguaje de programación completamente orientado a objetos, siguiendo el estándar de desarrollo, diagramación y codificación que estas ofrecen.

REFERENCIAS BIBLIOGRÁFICAS

- BOOCH, Grady. Análisis y Diseño Orientado a Objetos con Aplicaciones. Ed. Addison Wesley/ Díaz de Santos, USA, 1996
- BOOCH G., RUMBAUGHT J. y JACOBSON I. Lenguaje Unificado de Modelado. Ed. Pearson, 1999
- CHATAIN, Jean - Noel y DUSSAUCHOY, Alain. Sistemas Expertos. Métodos y Herramientas. Madrid: Ed. Paraninfo S.A., 1988
- FROUFE, Agustín. JAVA 2 Manual de Usuario y Tutorial. 2 ed. España: Ed. RA – MA , 2000
- GIARRATANO, Joseph y RILEY, Gary. Sistemas Expertos. Principios y Programación. 3ª. ed. México: Thomson Editores S.A., 2001
- HIDALGO, Luis Amador. Inteligencia Artificial y Sistemas Expertos. Córdoba: Ed. Servicio de Publicaciones Universidad de Córdoba, 1997
- LASALA CALLEJA, Pilar. Introducción a la Inteligencia Artificial y los Sistemas Expertos. Zaragoza: Prensas Universitarias de Zaragoza, 1994
- ROLSTON, David W. Principios de Inteligencia Artificial y Sistemas Expertos. México: Ed. Mc. Graw Hill, 1993
- WINSTON, Patrick. Inteligencia Artificial. México: Editorial Addison Wesley, 1994.