Sistemas Experto: Agencia de Viajes

Irene Marquet Gómez
Estudiante Ing. Telecomunicación
Universidad Carlos III de Madrid
100055841@alumnos.uc3m.es

Laura Luque Martínez
Estudiante Ing. Telecomunicación
Universidad Carlos III de Madrid
100047247@alumnos.uc3m.es

RESUMEN

Vamos a implementar un sistema experto con Clips cuyo propósito será aconsejar a un cliente sobre el viaje que mejor se ciñe a sus gustos y preferencias. Para ello primero haremos una breve descripción de los sistemas expertos y de su importancia y después pasaremos a desarrollar el problema con Clips, un sistema experto ampliamente utilizado.

Términos Generales Algorithms, Design

Palabras clave
Clips, sistema experto, agencia de viajes, árbol

1. INTRODUCION

Los sistemas expertos se pueden considerar como el primer producto verdaderamente operacional de la inteligencia artificial. Son programas de ordenador diseñados para actuar como un especialista humano en un dominio particular o área de conocimiento. En este sentido, pueden considerarse como intermediarios entre el experto humano, que transmite su conocimiento al sistema, y el usuario que lo utiliza para resolver un problema con la eficacia del especialista. El sistema experto utilizará para ello el conocimiento que tenga almacenado y algunos métodos de inferencia. Para que un sistema experto sea herramienta efectiva, los usuarios deben interactuar de una forma fácil, reuniendo dos capacidades para poder cumplirlo:

1. Explicar sus razonamientos o base del conocimiento: los sistemas expertos se deben realizar siguiendo ciertas reglas o pasos comprensibles de manera que se pueda generar la explicación para cada una de estas reglas, que a la vez se basan en hechos.

2. Adquisición de nuevos conocimientos o integrador del sistema: son mecanismos de razonamiento que sirven para modificar los conocimientos anteriores. Sobre la base de lo anterior se puede decir que los sistemas expertos son el producto de investigaciones en el campo de la inteligencia artificial ya que ésta no intenta sustituir a los expertos humanos, sino que se desea ayudarlos a realizar con más rapidez y eficacia todas las tareas que realiza.

1.1 Usos de un sistema experto

a) Un sistema experto es muy eficaz cuando tiene que analizar una gran cantidad de información, interpretándola y proporcionando una recomendación a partir de la misma. Un ejemplo es el análisis financiero, donde se estudian las oportunidades de inversión, dependiendo de los datos financieros de un cliente y de sus propósitos.

b) Para detectar y reparar fallos en equipos electrónicos, se utilizan los sistemas expertos de diagnóstico y depuración, que formulan listas de preguntas con las que obtienen los datos necesarios para llegar a una conclusión. Entonces recomiendan las acciones adecuadas para corregir los problemas descubiertos. Este tipo de sistemas se utilizan también en medicina (ej. MYCIN y PUFF), y para localizar problemas en sistemas informáticos grandes y complejos.

c) Los sistemas expertos son buenos para predecir resultados futuros a partir del conocimiento que tienen. Los sistemas meteorológicos y de inversión en bolsa son ejemplos de utilización en este sentido. El sistema PROSPECTOR es de este tipo.

d) La planificación es la secuencia de acciones necesaria para lograr una meta. Conseguir una buena planificación a largo plazo es muy difícil. Por ello, se usan sistemas expertos para gestionar proyectos de desarrollo, planes de producción de fábricas, estrategia militar y configuración de complejos sistemas informáticos, entre otros.

e) Cuando se necesita controlar un proceso tomando decisiones como respuesta a su estado y no existe una solución algorítmica adecuada, es necesario usar un sistema experto. Este campo comprende el supervisar fábricas automatizadas, factorías químicas o centrales nucleares. Estos sistemas son extraordinariamente críticos porque normalmente tienen que trabajar a tiempo real.

f) El diseño requiere una enorme cantidad de conocimientos debido a que hay que tener en cuenta muchas especificaciones y restricciones. En este caso, el sistema experto ayuda al diseñador a completar el diseño de forma competente y dentro de los límites de costes y de tiempo. Se diseñan circuitos electrónicos, circuitos integrados, tarjetas de circuito impreso, estructuras arquitectónicas, coches, piezas mecánicas, etc.

g) Por último, un sistema experto puede evaluar el nivel de conocimientos y comprensión de un estudiante, y ajustar el proceso de aprendizaje de acuerdo con sus necesidades.

1.2 Arquitectura y funcionamiento de un sistema experto

La mayoría de los sistemas expertos tienen unos componentes básicos: base de conocimientos, motor de inferencia, base de datos e interfaz con el usuario. Muchos tienen, además, un módulo de explicación y un módulo de adquisición del conocimiento.

1.2.1 Base de conocimientos

La base de conocimientos contiene el conocimiento especializado extraído del experto en el dominio. El método más común para representar el conocimiento es mediante reglas de producción. El dominio de conocimiento representado se divide, pues, en pequeñas fracciones de conocimiento o reglas. Una característica muy importante es que la base de conocimientos es independiente del mecanismo de inferencia que se utiliza para resolver los problemas. De esta forma, cuando los conocimientos almacenados se han quedado obsoletos, o cuando se dispone de *nuevos conocimientos, es relativamente fácil añadir reglas nuevas, eliminar las antiguas o corregir errores en las existentes.*

1.2.2 Base de datos

La base de datos o base de hechos es una parte de la memoria del ordenador que se utiliza para almacenar los datos recibidos inicialmente para la resolución de un problema. Contiene conocimiento sobre el caso concreto en que se trabaja. También se registrarán en ella las conclusiones intermedias y los datos generados en el proceso de inferencia.

1.2.3 Motor de inferencias

El motor de inferencias es un programa que controla el proceso de razonamiento que seguirá el sistema experto. Utilizando los datos que se le suministran, recorre la base de conocimientos para alcanzar una solución. La estrategia de control puede ser de encadenamiento progresivo o de encadenamiento regresivo. En el primer caso se comienza con los hechos disponibles en la base de datos, y se buscan reglas que satisfagan esos datos. Normalmente, el sistema sigue los siguientes pasos:

1. Evaluar las condiciones de todas las reglas respecto a la base de datos, identificando el conjunto de reglas que se pueden aplicar (aquellas que satisfacen su parte condición)

2. Si no se puede aplicar ninguna regla, se termina sin éxito; en caso contrario se elige cualquiera de las reglas aplicables y se ejecuta su parte acción (esto último genera nuevos hechos que se añaden a la base de datos)

3. Si se llega al objetivo, se ha resuelto el problema; en caso contrario, se vuelve al paso 1

Al encadenamiento regresivo se le suele llamar guiado por objetivos, ya que, el sistema comenzará por el objetivo (parte acción de las reglas) y operará retrocediendo para ver cómo se deduce ese objetivo partiendo de los datos.

1.2.4 Interfaz con el usuario

El interfaz de usuario permite que el usuario pueda describir el problema al sistema experto. Interpreta sus preguntas, los comandos y la información ofrecida. A la inversa, formula la información generada por el sistema incluyendo respuestas a las preguntas, explicaciones y justificaciones.

1.2.5 Modulo de explicacion

La mayoría de los sistemas expertos contienen un módulo de explicación, diseñado para aclarar al usuario la línea de razonamiento seguida en el proceso de inferencia. Si el usuario pregunta al sistema cómo ha alcanzado una conclusión, éste le presentará la secuencia completa de reglas usada

1.2.6 Modulo de adquisicion

El módulo de adquisición del conocimiento permite que se puedan añadir, eliminar o modificar elementos de conocimiento (en la mayoría de los casos reglas) en el sistema experto.

1.3 Ventajas e Inconvenientes de un Sistema experto

1.3.1 Ventajas

a) Un sistema experto mejora la productividad al resolver y decidir los problemas más rápidamente. Esto permite ahorrar tiempo y dinero. A veces sin esa rapidez las soluciones obtenidas serían inútiles.

b) Los valiosos conocimientos de un especialista se guardan y se difunden, de forma que, no se pierden aunque desaparezca el especialista.

c) Con un sistema experto se obtienen soluciones más fiables gracias al tratamiento automático de los datos, y más contrastadas, debido a que se suele tener informatizado el conocimiento de varios expertos.

d) Debido a la separación entre la base de conocimiento y el mecanismo de inferencia, los sistemas expertos tienen gran flexibilidad, lo que se traduce en una mejor modularidad, modificabilidad y legibilidad del conocimiento.

1.3.2 Inconvenientes

a) El conocimiento humano es complejo de extraer y, a veces, es problemático representarlo. Si un problema sobrepasa la competencia de un sistema experto, sus prestaciones se degradan de forma notable. Además, las estrategias de razonamiento de los motores de inferencia suelen estar programadas procedimentalmente y se adaptan mal a las circunstancias. Están limitados para tratar problemas con información incompleta.

b) Un experto humano no estudia progresivamente una hipótesis, sino que decide de inmediato cuando se enfrenta a una situación análoga a otra ocurrida en el pasado. Los sistemas expertos no utilizan este razonamiento por analogía.

c) Los costes y duración del desarrollo de un sistema experto son bastante considerables (aunque se suelen amortizar rápidamente) y su campo de aplicación actual es restringido y específico.

d) Problemas sociales que acarrean al ser susceptibles de influir en la estructura y número de empleos.

1.4 Clips

1.4.1 Introduccion a Clips

CLIPS es una herramienta que provee un ambiente de desarrollo para la producción y ejecución de sistemas expertos. Fue creado a partir de 1984, en el Lyndon B. Johnson Space Center de la NASA. Se trata de un acrónimo de C Language Integrated Production System (Sistema de Producción Integrado en Lenguaje C). En la actualidad, entre los paradigmas de programación que soporta CLIPS se encuentran la Programación lógica, la Programación imperativa y la Programación Orientada a Objetos. CLIPS es probablemente del sistema experto más ampliamente usado debido a que es rápido, eficiente y gratuito.

1.4.2 Caracteristicas de Clips

Las características principales de CLIPS son:

- Representación del Conocimiento: CLIPS permite manejar una amplia variedad de conocimiento, soportando tres paradigmas de programación: el declarativo, el imperativo, y el orientado a objetos.

- Portabilidad: CLIPS fue escrito en C con el fin de hacerlo más portable y rápido, y ha sido instalado en diversos sistemas operativos (Windows 95/98/NT, MacOS X, Unix) sin ser necesario modificar su código fuente. CLIPS puede ser ejecutado en cualquier sistema con un compilador ANSI de C, o un compilador de C++.

- Integrabilidad: CLIPS puede ser embebido en código imperativo, invocado como una sub-rutina, e integrado con lenguajes como C, Java, FORTRAN y otros.

- Desarrollo Interactivo: La versión estándar de CLIPS provee un ambiente de desarrollo interactivo y basado en texto; este incluye herramientas para la depuración, ayuda en línea, y un editor integrado.

- Verificación/Validación:
  CLIPS contiene funcionalidades que permiten verificar las reglas incluidas en el sistema experto que está siendo desarrollado.

- Documentación: En la página Web oficial de CLIPS se encuentra una extensa documentación que incluye un Manual de Referencia y una Guía del Usuario.

- Bajo Costo: CLIPS es un software de dominio público.

1.4.3 Derivadoss de Clips

Una de las razones del amplio uso de CLIPS está en sus derivados e interfaces con otros lenguajes, como:

- JESS: implementación de CLIPS en Java+.

- FuzzyCLIPS: incorpora a CLIPS la posibilidad de usar razonamiento difuso.

- CLIPSMM: una interfaz libre de CLIPS con C++.

- PHLIPS: extensión para PHP.

- EHSIS: Implementación del lenguaje CLIPS con APIs adicionales y documentación en castellano.

2. AGENCIA DE VIAJES CON CLIPS

2.1 Planteamiento y Diseño

Queremos realizar una agencia de viajes que nos sugiera diferentes destinos y actividades a realizar según lo requisitos demandados por el usuario. Para realizar este diseño hemos partido del diagrama de un árbol, en el que cada nodo tiene una pregunta asociada que se imprimirá por pantalla y que el cliente tendrá que responder, según la respuesta que responda el programa se encaminará a la rama del nodo correspondiente a la respuesta elegida por el cliente. Una vez recorrido todo el árbol se llegará a una de las hojas del árbol que contendrá la respuesta final.

Hemos planteado el diseño de manera que el árbol sea n-ario, es decir a que el árbol no se limite a tener como máximo dos nodos descendientes, así pues, cada nodo de nuestro árbol podrá tener dos o más respuestas asociadas a el.

A continuación le mostramos una imagen de la raíz del árbol, cuya pregunta asociada es la época del año en la que le gustaría viajar. Las cuatro ramas llevaran a replicas de nodos con las misma preguntas, la única diferencia será la respuesta final que según la época del año será diferente.

Vamos a describir el árbol de un ejemplo concreto, para otras opciones como ciudad o naturaleza, las preguntas que se realizarán al usuario serán diferentes.

Figura 1

Según se elija playa, naturaleza o ciudad se ira a un nodo diferente cuya pregunta asociada será distinta. A continuación se muestran los tres árboles correspondientes a la elección de playa, ciudad y naturaleza.

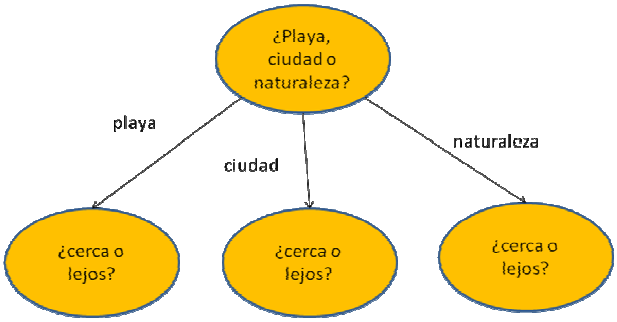

Figura 2

Si por ejemplo hemos elegido otoño, descendemos por el nodo hijo de otoño que toma la decisión sobre ¿playa, ciudad o naturaleza? Y según la decisión que tomemos sobre cerca o lejos, iremos por la rama correspondiente.

La siguiente decisión a tomar está representada por el siguiente esquema:

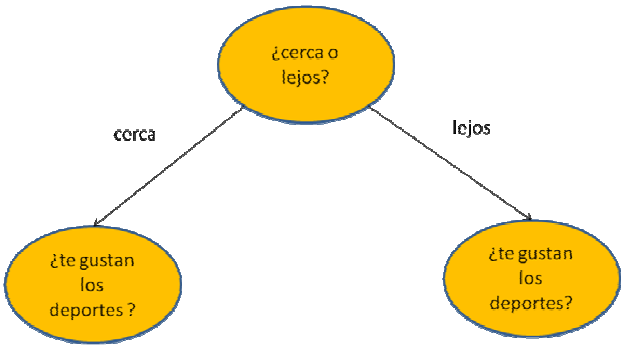

Figura 3

Para diferenciar cada tipo de nodo y la decisión que implica ese nodo, utilizamos el atributo type definido en node, de este modo conseguimos tratar de diferente manera un nodo que decide sobre la estación del año en que se quiere viajar y que tiene cuatro hijos, de un nodo que decide sobre si se quiere viajar cerca o lejos y que solo tiene dos hijos.

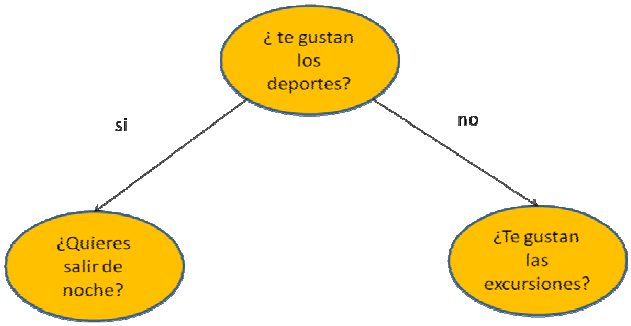

Figura 4

Si decidimos que no queremos deportes se nos ofrece la posibilidad de ir a excursiones, y si nos decantamos por la opción de los deportes se nos pregunta si queremos salir de fiesta.

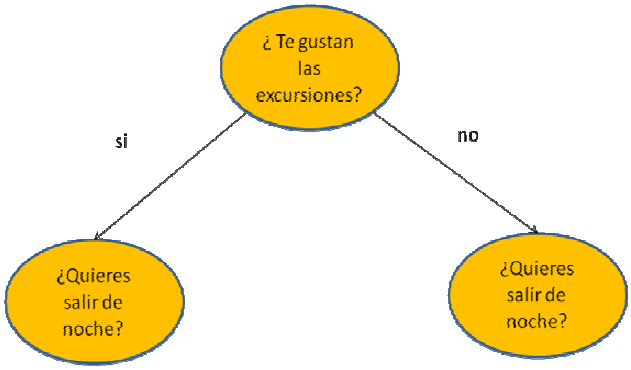

Figura 5

Para las dos opciones se pregunta también si se desea salir de noche. Si hubiésemos seguido por el nodo de deportes hubiésemos llegado a un nodo con el mismo tipo de decisión que en este caso. Cuando respondemos a la pregunta sobre salir de noche, llegamos al nodo final que nos muestra los destinos y las propuestas adecuadas a la cadena de respuestas introducidas por el usuario.

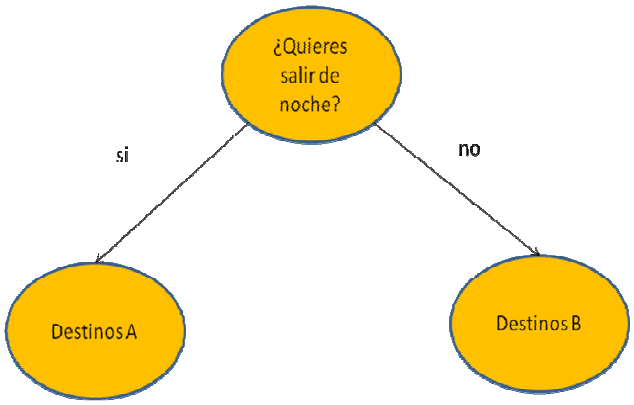

Figura 6

2.2 Implementación

Para implementar nuestro árbol en clips dividimos nuestro código en tres grandes bloques:

2.2.1 Definicion de Plantillas

Las plantillas son como clases pero sin herencia. Nosotros definimos una plantilla que contendrá como atributos todos los nodos de nuestro árbol, el nombre del nodo, el tipo de nodo (pregunta o respuesta), la pregunta correspondiente al nodo, y la respuesta correspondiente al nodo:

```
(deftemplate node
(slot name)
(slot type)
(slot question)
(slot otonyo-node)
(slot invierno-node)
;;;Definimos aquí todos los nodos del árbol
(slot answer))
```

2.2.2 Motor de inferencias

El motor de inferencias trata de emparejar la lista de hechos con los patrones de las reglas. En este bloque definimos las reglas encargadas de inicializar el programa, finalizar el programa, realizar las preguntas, obtener el valor de las respuestas, decidir cual es el siguiente nodo según el nodo en el que se este y la respuesta del cliente, de asegurarse de si la respuesta dada por teclado es valida, y de preguntar al cliente si la respuesta le satisface.

En el siguiente ejemplo vemos la regla encargada de preguntar si se desea un viaje asociado con la playa, ciudad o con la naturaleza; de ver si la respuesta dada por teclado es correcta y según responda el cliente dirigirse al nodo asociado a la respuesta dada:

```
;;realiza la pregunta
(defrule ask-decision-paisaje-node-question
?node <- (current-node ?name)
(node (name ?name)
(type decisionpaisaje)
(question ?question))
(not (answer ?))
=>
(printout t ?question " (playa, ciudad o naturaleza) ")
(assert (answer (read))))
;;ver si la decision es correcta
(defrule bad-answer-paisaje
?answer <- (answer ~playa&~ciudad&~naturaleza)
=>
(retract ?answer))
;;playa
(defrule proceed-to-playa-branch
?node <- (current-node ?name)
(node (name ?name)
(type decisionpaisaje)
(playa-node ?playa-branch))
?answer <- (answer playa)
=>
(retract ?node ?answer)
(assert (current-node ?playa-branch)))
;;ciudad
(defrule proceed-to-ciudad-branch
node <- (current-node ?name)
(node (name ?name)
(type decisionpaisaje)
(ciudad-node ?ciudad-branch))
?answer <- (answer ciudad)
=>
(retract ?node ?answer)
(assert (current-node ?ciudad-branch)))
;;naturaleza
(defrule proceed-to-naturaleza-branch
?node <- (current-node ?name)
(node (name ?name)
(type decisionpaisaje)
(naturaleza-node ?naturaleza-branch))
?answer <- (answer naturaleza)
=>
(retract ?node ?answer)
(assert (current-node ?naturaleza-branch)))
```

En el anterior ejemplo se pueden observar las cinco reglas asociadas a la pregunta: "¿desea playa, ciudad o naturaleza?".

En este ejemplo tenemos tres posibles ramas por las que podemos descender en el árbol, por ello definimos tres saltos diferentes según las respuesta que haya introducido el usuario.

2.2.3 Archivo destinos.dat

En este archivo nos encargamos de definir todos los nodos del programa. En el siguiente ejemplo vemos como se define el nodo raíz que contiene varios atributos: el nombre del nodo actual (i.e.root), el tipo de decisión, la pregunta que se formula en dicho nodo, los nodos asociados a su respuesta, y la respuesta asociada a este nodo que en este caso es nil:

(node (name root) (type decision) (question "En que estacion le gustaria viajar?") (otonyo-node node1) (invierno-node node2) (primavera-node node3)(verano-node node4)(answer nil))

3. EJECUCIÓN

Esta es una ejcución de ejemplo en la que las elecciones son: invierno – playa – lejos – con deportes – salir de noche .
Al elegir la estación de invierno se recomienda al usuario un destino con el clima adecuado para esa época elegida.

En que estacion le gustaria viajar? (otonyo, invierno, verano, primavera) invierno

Que le apetece? (playa, ciudad o naturaleza) playa

Quiere irse... (cerca o lejos) lejos

Le gustan los deportes? (si o no) si

Te gustaria salir de noche? (si o no) si

LE RECOMIENDO:

Punta Cana,(Rep.Dominicana): Cursos de buceo, actividades de futbol-playa, voley-playa,etc. y salidas a discotecas: Mangu, Areito,etc.

/ Cancun,(mexico): Cursos de buceo, actividades de futbol-playa, voley-playa,etc. y salidas a discotecas: Coco-Bongo, Glazz,etc.

Le gusta alguno de estos destinos? (si o no) si

Otra prueba? (si o no) no

Veamos otro ejemplo de ejecución para el caso en el que se elige: verano – ciudad – fuera de europa – occidental – compras.

En que estacion le gustaria viajar? (otonyo, invierno, verano, primavera) verano

Que le apetece? (playa, ciudad o naturaleza) ciudad

Desea irse a Europa o fuera de Europa?... (Europa o fuera) fuera

Te llama mas la cultura oriental o la occidental (oriental u occidental) occidental

Te gusta mas disfrutar de... (compras o arte) compras

LE RECOMIENDO:

Nueva York,(EEUU) / Los Angeles,(EEUU) / Las Vegas,(EEUU) / Sidney,(Australia)

Le gusta alguno de estos destinos? (si o no) si

Otra prueba? (si o no) no

Una ejecución de ejemplo para el caso en el que nos desviamos por la rama de naturaleza, eligiendo: primavera – naturaleza – fuera de Europa – calor – desierto.

En que estacion le gustaria viajar? (otonyo, invierno, verano, primavera) primavera

Que le apetece? (playa, ciudad o naturaleza) naturaleza

Desea irse a Europa o fuera de Europa?... (Europa o fuera) fuera

Quieres un ambiente con... (frio o calor) calor

Quieres un paisaje con... (desierto o montanya) desierto

LE RECOMIENDO:

El Cairo,(Egipto) / La Guajira,(Colombia) / Arizona(EEUU)

Le gusta alguno de estos destinos? (si o no) si

Otra prueba? (si o no) no

Una ejecución de ejemplo para el caso en el que nos desviamos por la rama de ciudad, eligiendo: verano – ciudad – Europa – norte – relajacion.

En que estacion le gustaria viajar? (otonyo, invierno, verano, primavera) verano

Que le apetece? (playa, ciudad o naturaleza) ciudad

Desea irse a Europa o fuera de Europa?... (Europa o fuera) Europa

Prefieres el centro de Europa o el norte? (central o norte) norte

Que prefieres tomarte un tiempo de .... (relajacion o deportes) relajacion

LE RECOMIENDO:

Helsinki,(Finladia) / Malmo,(Suecia)

Le gusta alguno de estos destinos? (si o no) si

Otra prueba? (si o no) no

Y por último un ejemplo si nos queremos ir fuera de Europa a ver arte, eligiendo: verano – ciudad – fuera de Europa – occidental – arte.

En que estacion le gustaria viajar? (otonyo, invierno, verano, primavera) invierno

Que le apetece? (playa, ciudad o naturaleza) ciudad

Desea irse a Europa o fuera de Europa?... (Europa o fuera) fuera
Te llama mas la cultura oriental o la occidental (oriental u occidental) occidental

Te gusta mas disfrutar de... (compras o arte) arte

LE RECOMIENDO:

Cordoba,(Argentina) / Buenos Aires,(Argentina) / Ciudad de Mexico,(Mexico)

Le gusta alguno de estos destinos? (si o no) si

Otra prueba? (si o no) no

4. REFERENCIAS

[1] http://es.wikipedia.org/wiki/Sistema_experto

[2] http://www.monografias.com/trabajos10/exper/exper.shtml

[3] http://es.wikipedia.org/wiki/CLIPS

[4] http://www.redcientifica.com/doc/doc199908210001.html

[5] http://wwwdi.ujaen.es/~dofer/ico/material/CLIPS-Tutorial-1.html

[6] http://wwwdi.ujaen.es/~dofer/ico/material/CLIPS-Tutorial-2.html

[7] Apuntes de la asignatura