

***Tot***

*Diseño*

Grupo Desarrollador:  
Ivan Araya  
Jorge Campos

## Revisión

Versión	Fecha	Resumen del cambio	Autor
1	15/10/2016	Primera versión documento	Iván Araya Jorge Campos
2	20/11/2016	Segunda versión documento	Iván Araya Jorge Campos
3	08/07/16	Tercera versión documento	Iván Araya Jorge Campos

## **Contenido**

### 1 Descripción del Modelo Solución

### 2 Descripción del Modelo Solución

### 3 Módulo: Librería sistema experto

#### 3.1 Arquitectura de la librería

##### 3.1.1 Descripción de comunicaciones

#### 3.2 Diseño de Datos

##### 3.2.1 Estructura Variable :

##### 3.2.2 Estructura Hecho:

##### 3.2.3 Estructura Regla:

##### 3.2.4 Estructura MetadatosBaseDeConocimiento:

#### 3.3 Diagrama de clases Libreria Sistema Experto

#### 3.4 Sub módulo gestión del conocimiento

#### 3.5 Sub módulo Motor de inferencia

#### 3.6 Diseño de interfaz

##### 3.6.1 GestionadorBaseDeConocimiento

##### 3.6.2 Encadenamiento hacia adelante

##### 3.6.3 Encadenamiento hacia atrás

##### 3.6.4 Lectura de la base de conocimiento

#### 3.7 Diseño procedimental

##### 3.7.1 Encadenamiento hacia atrás

##### 3.7.2 Encadenamiento hacia adelante

### 4 Módulo: Aplicación Tot IDE

#### 4.1 Arquitectura del Sistema

#### 4.2 Diagrama de Componentes

#### 4.3 Diseño de Interfaz

##### 4.3.1 Interfaz de Usuario

##### 4.3.2 Ventana principal

##### 4.3.3 Ventana configuración

##### 4.3.4 Ventana Gestión de la base de conocimiento

###### 4.3.4.1 Panel variables

###### 4.3.4.2 Panel Hechos

###### 4.3.4.3 Panel Reglas

##### 4.3.5 Ventana configuración log de inferencia

##### 4.3.6 Ventana configuración log de inferencia

##### 4.3.7 Consulta Sistema experto

###### 4.3.7.1 Dialogo inicial

---

[4.3.7.2 Dialogo selección de objetivo](#)

[4.3.7.3 Dialogo Consulta de usuario](#)

[4.3.7.4 Dialogo validación de regla](#)

[4.3.7.5 Dialogo validación de regla](#)

[4.3.7.6 Dialogo guardar log](#)

## [5 Módulo: Aplicación Tot Consultas sistema experto](#)

### [5.1 Arquitectura del Sistema](#)

#### [5.1.1 Diagrama de Componentes](#)

### [4.3 Diseño de Interfaz](#)

#### [4.3.1 Interfaz de Usuario](#)

#### [4.3.2 Ventana principal](#)

#### [4.3.5 Ventana configuración log de inferencia](#)

#### [4.3.7 Consulta Sistema experto](#)

##### [4.3.7.1 Dialogo inicial](#)

##### [4.3.7.2 Dialogo selección de objetivo](#)

##### [4.3.7.3 Dialogo Consulta de usuario](#)

##### [4.3.7.4 Dialogo validación de regla](#)

##### [4.3.7.5 Dialogo validación de regla](#)

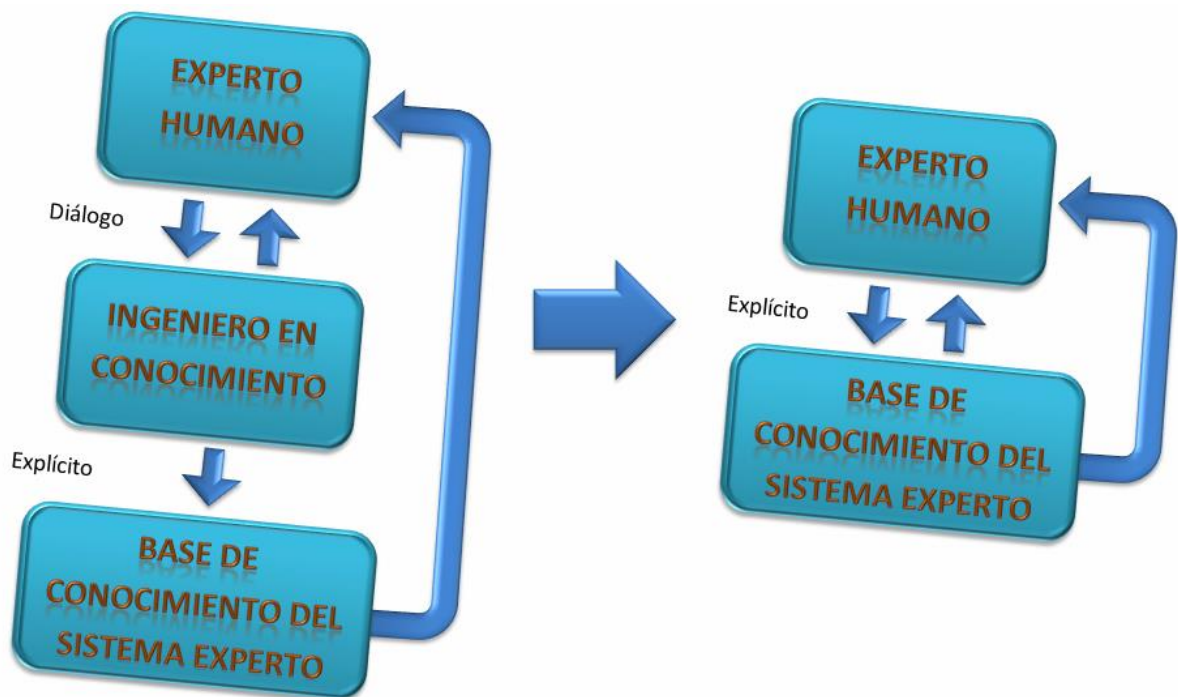
## 1. Descripción del Modelo Solución

Hoy en día se ha construido una gran variedad de sistemas expertos basados en el conocimiento. grandes sistemas que contienen miles de reglas (Ej. XCON/R1 de Digital Equipment Corporation, asistencia de pedidos de computadores) u otros con contenido más especializado con unos cientos de reglas.

Un sistema experto clásico abarca el conocimiento no escrito que debe obtenerse de un especialista a través de extensas entrevistas con un ingeniero del conocimiento, durante un largo periodo de tiempo, mediante el proceso de ingeniería del conocimiento en donde se adquiere el conocimiento de un experto humano o de otra fuente y se codifica en el sistema experto. En la parte izquierda de la imagen a continuación se puede ver el flujo de trabajo normal del proceso de ingeniería de conocimiento.

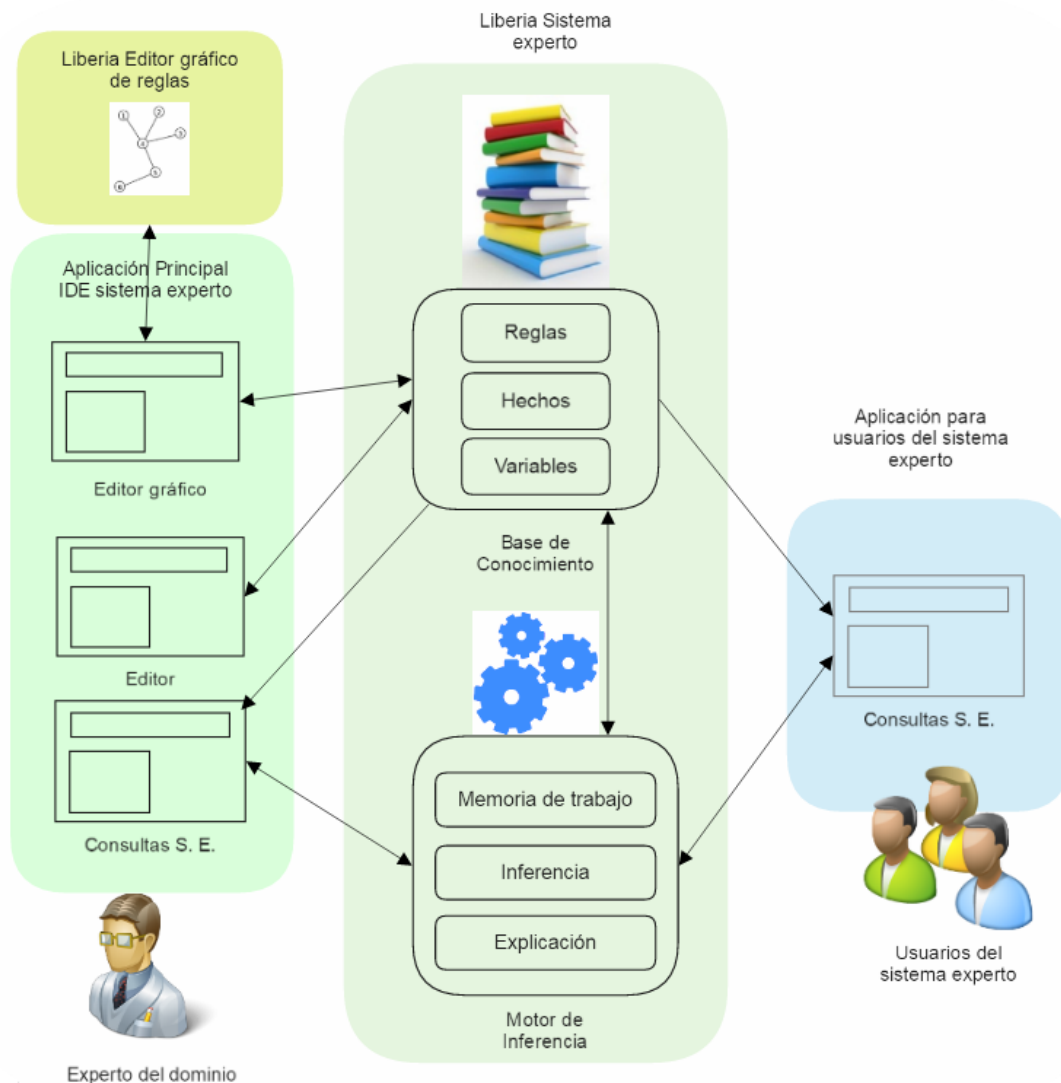
Entonces dentro de la problemática de la ingeniería del conocimiento podemos destacar dos problemas principales, el tiempo y la complejidad que le requiere al ingeniero del conocimiento entender el problema para poder codificar las reglas y por otra parte el hecho de que el experto a consultar no va a tener todo el conocimiento presente en un instante dado.

Es por esto que la creación de una herramienta en donde el propio experto pueda ir ingresando en el tiempo su conocimiento (parte derecha imagen), permitirá tener un sistema experto en menor tiempo, lo cual dejaría al ingeniero del conocimiento como un supervisor de la base de conocimiento.





## 2. Descripción del Modelo Solución



El modelo solución final para satisfacer las necesidades del proyecto consta de los siguientes elementos:

- **Aplicación principal:** Esta es la aplicación utilizada por el experto del dominio en donde podrá hacer la gestión general y realizar consultas a la base de conocimiento.
- **Librería sistema experto:** Aquí se encuentran implementadas las funcionalidades necesarias para la gestión de la base de conocimiento y la realización de procesos de inferencia, por parte del usuario desarrollador.
- **Aplicación para usuarios del sistema experto:** Aplicación que tiene como objetivo a los usuarios finales del sistema experto creado por el experto de dominio, permitiendo la realización de consultas a la base de conocimiento pero sin hacer edición de esta.

---

## 3. Módulo: Librería sistema experto

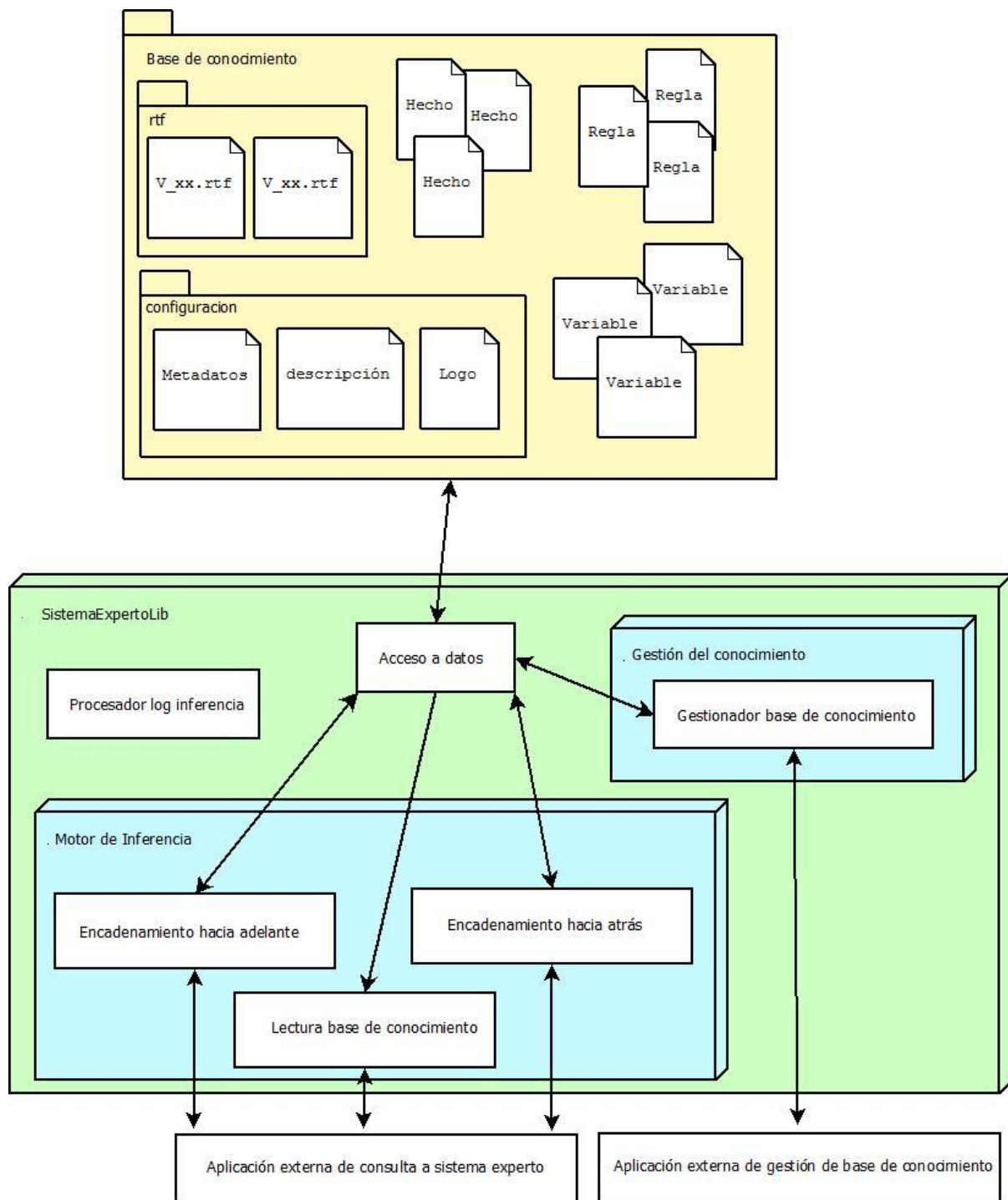
### 3.1. Arquitectura de la librería

Este módulo provee de proveer de las estructuras de datos y de las funciones necesarias para hacer gestión de la base de conocimiento y realizar procesos de inferencia.

Características generales del módulo:

- Se cuenta con tres tipos de estructuras principales que se utilizan el modelamiento de la base de conocimiento:
  - Variable
  - Hecho
  - Regla
- Estas estructuras dependen unas de las otras en el siguiente orden
  - Regla puede contener de 1..n hechos en el antecedente y 1 hecho en el consecuente.
  - Hecho contiene 1 variable en su definición
  - Variable no tiene dependencias.
- Cada variable puede tener asociado un archivo rtf, para guardar su descripción.
- Existe una estructura gestionada por la librería llamada MetadatosBaseDeConocimiento que guarda la información general de la base de conocimiento.
- Las estructuras Regla, Hecho, Variable y metadatos se guardan de forma binaria.
- La creación, modificación y eliminación de cualquier elemento de la base de conocimiento puede hacerse sólo a través del submódulo GestiónDelConocimiento.
- El submódulo de inferencia cuenta con una clase para leer la base de conocimiento, sin realizar cambios.
- El proceso de inferencia se lleva a cabo a través de las clases EncadenamientoHaciaAdelante y EncadenamientoHaciaAtras
- El módulo cuenta con una clase para procesar el log resultante de la inferencia llamada ProcesadorLogInferencia.





### 3.1.1. Descripción de comunicaciones

#### **Base de conocimiento:**

**Acceso a datos:** provee el repositorio para el almacenamiento y la lectura de los distintos elementos de la base de conocimiento

#### **Acceso a datos:**

**Base de conocimiento:** Realiza los procesos de lectura, escritura, modificación y eliminación de los archivos de la base de conocimiento.

**Gestionador base de conocimiento:** Envía las distintas instanciaciones de datos para ser utilizadas en la edición de la base de conocimiento..

**Lectura base de conocimiento:** Envía las instanciaciones de datos para ser leídas externamente.

**Encadenamiento hacia adelante:** Envía las distintas instanciaciones de regla, hechos, variables y los metadatos datos para ser utilizadas en el proceso de inferencia.

**Encadenamiento hacia atrás:** Envía las distintas instanciaciones de regla, hechos, variables y los metadatos datos para ser utilizadas en el proceso de inferencia.

#### **Gestionador base de conocimiento:**

**Acceso a datos:** Solicita las instanciaciones de datos, envía las instancias a ser ingresadas, modificadas o eliminadas.

**Aplicación externa Editor:** Envía la instanciación de datos para ser mostradas al usuario.

#### **Encadenamiento hacia atrás o adelante:**

**Acceso a datos:** Envía las instancias de regla, hecho y variable a ser inicializadas o modificadas en sus valores. Solicita los metadatos de la base de conocimiento.

**Aplicación externa:** Solicita mediante eventos los valores de una variable preguntable al usuario, la validación de hechos inferidos y datos de control para la continuación o el término de la inferencia.

#### **Lectura base de conocimiento:**

**Acceso a datos:** solicita información de las instanciaciones de Variables, Hechos, Reglas y los metadatos de la base de conocimiento.

**Aplicación externa Consulta:** Envía la información de las instanciaciones de Variables, Hechos, Reglas y los metadatos de la base de conocimiento, para su lectura externa al módulo.

#### **Aplicación externa Editor:**

**Gestionador base de conocimiento:** Solicita información de las instanciaciones de Variables, Hechos, Reglas y los metadatos de la base de conocimiento, envía información a ser comprobada e ingresada por el gestor.

**Aplicación externa Consulta:**

**Encadenamiento hacia atrás o adelante:** Responde a los eventos de solicitud información al usuario de una variable, validación de hechos y el control de la inferencia.

**Lectura base de conocimiento:** Solicita información de las instanciaciones de Variables, Hechos, Reglas para ser utilizadas tanto en los eventos de información al usuario como en el registro de log de la inferencia.

## 3.2. Diseño de Datos

- La librería utiliza principalmente 4 estructuras de datos:
  - Variable
  - Hecho
  - Regla
  - MetadatosBaseDeConocimiento

### 3.2.1. Estructura Variable :

La estructura variable es la base del conocimiento puede ser de 3 tipos:

- Booleana: Sólo admite estados de true o false;
- Numérico: Admite entradas numéricas definidas como Real o Cardinal, además se puede establecer un rango para su estado.
- Lista: Admite cualquier estado de una lista predefinida por el usuario.

A continuación se muestra en detalle los atributos y métodos públicos de la estructura.

Constantes	
const int BOOLEANO	Tipo de variable BOOLEANO
const int NUMERICO	Tipo de variable NUMERICO
const int LISTA	Tipo de variable LISTA
Atributos	
string id_variable	Id de la variable
string nombre_variable	Nombre de la variable
int tipo_variable	Tipo de variable según constantes BOOL, NUMERICO, LISTA
object valor_variable	Obtiene el valor de la variable según su tipo
bool valor_booleano_actual	Valor BOOLEANO actual en la memoria de trabajo

double valor_numerico_actual	Valor NUMERICO actual en la memoria de trabajo
string valor_lista_actual	Valor LISTA actual en la memoria de trabajo
bool variable_modificada	Indica si el valor de la variable ha sido modificado
bool cardinal	Valor que establece la Variable como Cardinal (solo si la variable es de tipo numérica)
bool rango_limitado	Si la variable es de tipo numérica, la propiedad especifica si tiene un rango limitado
double rango_limite_inferior	Número menor que puede tomar la variable (Si la variable es numérica y con rango limitado)
double rango_limite_superior	Número mayor que puede tomar la variable (Si la variable es numérica y con rango limitado)
string texto_consulta_variable	Texto con el cual se consultará la variable del sistema experto
string ruta_texto_descriptivo	Ruta Texto descriptivo de la variable
bool variable_de_inicio	Establece si la variable es solicitado al inicio de la inferencia
public bool variable_preguntable_al_usuario	Establece si el variable es preguntable al usuario
public bool variable_objetivo	Establece si el variable es objetivo inicial del encadenamiento hacia atrás
bool chequeo_de_consistencia	Atributo que indica si la variable es consistente en la base de conocimiento
<b>Métodos</b>	
Variable()	Constructor
Variable (string id_variable)	Constructor
agregarElementoALista(string elemento)	Método para agregar elementos a las opciones de la variable
eliminarElemento(string elemento)	Método para eliminar un elemento de la lista de opciones
elementoExistente (string elemento)	Método que comprueba la existencia de un elemento en la lista de opciones
limitarRangoVariable(double limite_inferior, double limite_superior)	Método que establece un rango a la variable de tipo NUMERICO
quitarRangosVariables()	Método que quita el rango establecido para la variable
Método que lista las Opciones de la variable	listarOpciones()
limpiarVariableParaInferencia()	Método que limpia los atributos asociados al proceso de inferencia

### 3.2.2. Estructura Hecho:

La estructura Hecho es la encargada de asociar una variable a una condición para ser evaluada posteriormente todo hecho adquiere el tipo de la variable asociada. A continuación se muestra en detalle los atributos y métodos públicos de la estructura.

Constantes	
const int BOOLEANO = 1	Tipo de hecho asociado a la variable
const int NUMERICO = 2	Tipo de hecho asociado a la variable
const int LISTA = 3	Tipo de hecho asociado a la variable
static string[] OPCIONES_BOOLEANO	Opciones de condición para tipo de hecho BOOLEANO{"FALSO","VERDADERO"}
static string[] OPCIONES_NUMERICO	Opciones de condición para tipo de hecho NUMERICO ["MENOR","MENOR O IGUAL","IGUAL","MAYOR O IGUAL", "MAYOR"]
static string[] OPCIONES_LISTA	Opciones de condición para tipo de hecho LISTA{"ES","NO ES"}
Atributos	
string id_hecho	identificador del Hecho
string id_variable	identificador de la variable utilizada en el hecho
int tipo_variable	Tipo de variable según constantes BOOL,NUMERICO, LISTA
string nombre_variable	nombre de la variable
string condicion	identificador de la variable utilizada en el hecho
bool valor_booleano	Valor booleano del hecho en la condición
string valor_lista_hecho	Valor Lista del hecho en la condición
double valor_numerico	Valor Numerico del hecho en la condición
bool hecho_preguntable_al_usuario	Obtiene si el hecho es preguntable al usuario
bool hecho_objetivo	Obtiene si el hecho es objetivo del encadenamiento hacia atrás
bool hecho_evaluado	Indica si el hecho fue evaluado en su condición, por la inferencia
bool estado_hecho	Si el hecho está evaluado, indica TRUE si cumple la condición FALSE en caso contrario
bool chequeo_de_consistencia	Atributo que indica si el hecho es consistente en la base de conocimiento
Métodos	

Hecho(string id_hecho, Variable variable_hecho)	Constructor
void actualizarParametrosVariableHecho(Variable variable_hecho)	Método que actualiza los parámetros del hecho en caso de que cambios en la variable
void establecerCondicion(string opcion_condicion)	Establece la condición para el tipo de hecho BOOLEANO
void establecerCondicion(string opcion_condicion, double valor)	Establece la condición para el tipo de hecho NUMERICO
void establecerCondicion(string opcion_condicion, string valor_en_variable)	Establece la condición para el tipo de hecho LISTA
bool comprobarCondicion(string opcion_condicion)	Método que comprueba la condición según el tipo de variable del objeto
bool evaluarHecho(object estado_variable)	Método para evaluar el estado de un hecho
bool evaluarHechoBooleano(bool estado_variable)	Método para evaluar tipo de hecho con variable booleana
bool evaluarHechoNumerico(double estado_variable)	Método para evaluar tipo de hecho con variable numérica
bool evaluarHechoLista(string estado_variable)	Método para evaluar un hecho con tipo de variable LISTA
void limpiarEstadoHechoParaInferencia ()	Método que limpiar los atributos del hecho utilizados en la influencia
override bool Equals(object obj)	Método Equals modificado para comparar en base a Variable, condición y valor en la condición
override string ToString()	VARIABLE + CONDICION + VALOR

### 3.2.3. Estructura Regla:

La estructura Regla agrupa los Hechos para formar el conocimiento ingresado por el experto y son la base del proceso de inferencia.

A continuación se muestra en detalle los atributos y métodos públicos de la estructura.

Atributos	
string id_regla	Identificador unico de la regla
string id_hecho_consecuente	ID del hecho consecuente de la regla
bool regla_validada	Indica si la regla a sido validada por el usuario en el proceso de inferencia, una regla no puede ser validada si no tiene todos los estados de los antecedentes en TRUE
bool chequeo_de_consistencia	Atributo que indica si la regla es consistente en la base de conocimiento

Métodos	
Regla(string id_regla)	Constructor
bool agregarHechoAlAntecedente(Hecho hecho)	Se agrega un nuevo Hecho a lista de antecedentes
void agregarHechoAlConsecuente(Hecho hecho)	Establece un nuevo Hecho en el consecuentes
bool consultarAntecedente(string id_hecho)	Método que muestra si el Hecho se encuentra en la lista de antecedentes
void cambiarEstadoHecho(string id_hecho, bool estado_hecho)	Método que cambia el estado del hecho en la regla ya sea en antecedente o consecuente
bool consultarConsecuente(string id_hecho)	Método que consulta si el hecho es el antecedente
void eliminarAntecedente(string id_hecho)	Método que elimina un hecho de los antecedentes
void eliminarConsecuente()	Método que elimina el hecho consecuente de la regla
string[] listarAntecedentes()	Método que lista los hechos antecedentes
string[] listarAntecedentesNoEstablecidos()	Método que lista los hechos no establecidos de la regla
void limpiarReglaParaInferencia()	Método que limpia los atributos asociados al proceso de inferencia
override string ToString()	ToString modificado
override bool Equals(object obj)	Equals modificado, compara dos reglas en base a sus hechos sin importar la ID

### 3.2.4. Estructura MetadatosBaseDeConocimiento:

Esta estructura sirve para guardar la información de configuración de la base de conocimiento.

Constantes	
const int ENCADENAMIENTO_HACIA_ATRAS = 1	Constante que indica el tipo de encadenamiento a utilizar en la base de conocimiento
const int ENCADENAMIENTO_HACIA_ADELANTE = 2	Constante que indica el tipo de encadenamiento a utilizar en la base de conocimiento
Atributos	
string titulo_sistema_experto	Obtiene o establece el título del sistema experto
string ruta_rtf_descripcion_sistema_experto	Obtiene o establece la ruta del archivo rtf descriptivo del sistema experto
string ruta_imagen_logo_sistema_experto	Obtiene o establece la ruta del logo del sistema experto

int tipo_de_encadenamiento	Obtiene o establece el tipo de encadenamiento de la base de conocimiento según las constantes ENCADENAMIENTO_HACIA_ATRAS ,ENCADENAMIENTO_HACIA_ADELANTE
bool base_conocimiento_chequeada	Obtiene o establece si la base de conocimiento fue chequeada por el gestor encargado





### 3.3. Diagrama de clases Libreria Sistema Experto

**Variable**  
Clase

**Campos**

- \_cardinal : bool
- \_chequeo\_de\_consistencia : bool
- \_id\_variable : string
- \_nombre\_variable : string
- \_rango\_limitado : bool
- \_rango\_limite\_inferior : double
- \_rango\_limite\_superior : double
- \_ruta\_imagen\_descriptiva : string
- \_ruta\_texto\_descriptivo : string
- \_texto\_consulta\_variable : string
- \_tipo\_variable : int
- \_valor\_booleano\_actual : bool
- \_valor\_lista\_actual : string
- \_valor\_numerico\_actual : double
- \_variable\_de\_inicio : bool
- \_variable\_modificada : bool
- \_variable\_objetivo : bool
- \_variable\_preguntable\_al\_usuario : bool
- BOOLEANO : int
- LISTA : int
- lista\_de\_opciones : ArrayList
- NUMERICO : int

**Propiedades**

- cardinal : bool
- chequeo\_de\_consistencia : bool
- id\_variable : string
- nombre\_variable : string
- rango\_limitado : bool
- rango\_limite\_inferior : double
- rango\_limite\_superior : double
- ruta\_imagen\_descriptiva : string
- ruta\_texto\_descriptivo : string
- texto\_consulta\_variable : string
- tipo\_variable : int
- valor\_booleano\_actual : bool
- valor\_lista\_actual : string
- valor\_numerico\_actual : double
- valor\_variable : object
- variable\_de\_inicio : bool
- variable\_modificada : bool
- variable\_objetivo : bool
- variable\_preguntable\_al\_usuario : bool

**Métodos**

- agregarElementoALista() : void
- elementoExistente() : bool
- eliminarElemento() : void
- limitarRangoVariable() : void
- limpiarVariableParaInferencia() : void
- listarOpciones() : string[]
- quitarRangosVariables() : void
- Variable() (+ 1 sobrecarga)

**Hecho**  
Clase

**Campos**

- \_chequeo\_de\_consistencia : bool
- \_condicion : string
- \_estado\_hecho : bool
- \_hecho\_evaluado : bool
- \_hecho\_objetivo : bool
- \_hecho\_preguntable\_al\_usuario : bool
- \_id\_hecho : string
- \_id\_variable : string
- \_nombre\_variable : string
- \_tipo\_variable : int
- \_valor\_booleano\_hecho : bool
- \_valor\_lista\_hecho : string
- \_valor\_numerico\_hecho : double
- BOOLEANO : int
- LISTA : int
- NUMERICO : int
- OPCIONES\_BOOLEANO : string[]
- OPCIONES\_LISTA : string[]
- OPCIONES\_NUMERICO : string[]

**Propiedades**

- chequeo\_de\_consistencia : bool
- condicion : string
- estado\_hecho : bool
- hecho\_evaluado : bool
- hecho\_objetivo : bool
- hecho\_preguntable\_al\_usuario : bool
- id\_hecho : string
- id\_variable : string
- nombre\_variable : string
- tipo\_variable : int
- valor\_booleano : bool
- valor\_lista\_hecho : string
- valor\_numerico : double

**Métodos**

- actualizarParametrosVariableHecho() : void
- comprobarCondicion() : bool
- Equals() : bool
- establecerCondicion() : void (+ 2 sobrecargas)
- evaluarHecho() : bool
- evaluarHechoBooleano() : bool
- evaluarHechoLista() : bool
- evaluarHechoNumerico() : bool
- GetHashCode() : int
- Hecho()
- limpiarEstadoHechoParaInferencia() : void
- ToString() : string

**Regla**  
Clase

**Campos**

- \_chequeo\_de\_consistencia : bool
- \_id\_regla : string
- \_regla\_validada : bool
- antecedentes : ArrayList
- consecuente : DatosHechos

**Propiedades**

- chequeo\_de\_consistencia : bool
- id\_hecho\_consecuente : string
- id\_regla : string
- regla\_validada : bool

**Métodos**

- agregarHechoAlAntecedente() : bool
- agregarHechoAlConsecuente() : void
- cambiarEstadoHecho() : void
- consultarAntecedente() : bool
- consultarConsecuente() : bool
- eliminarAntecedente() : void
- eliminarConsecuente() : void
- Equals() : bool
- GetHashCode() : int
- limpiarReglaParaInferencia() : void
- listarAntecedentes() : string[]
- listarAntecedentesNoEstablecidos() : string[]
- Regla()
- ToString() : string

**Tipos anidados**

**DatosHechos**  
Struct

**Campos**

- estado\_hecho : bool
- hecho\_establecido : bool
- id\_hecho : string
- nombre\_hecho : string

**MetadatosBaseDeConocimiento**  
Clase

**Campos**

- \_base\_conocimiento\_chequeada : bool
- \_ruta\_imagen\_logo\_sistema\_experto : string
- \_ruta\_rtf\_descripcion\_sistema\_experto : string
- \_tipo\_de\_encadenamiento : int
- \_titulo\_sistema\_experto : string
- ENCADENAMIENTO\_HACIA\_ADELANTE : int
- ENCADENAMIENTO\_HACIA\_ATRAS : int

**Propiedades**

- base\_conocimiento\_chequeada : bool
- ruta\_imagen\_logo\_sistema\_experto : string
- ruta\_rtf\_descripcion\_sistema\_experto : string
- tipo\_de\_encadenamiento : int
- titulo\_sistema\_experto : string

**ProcesadorLogInferencia**  
 Clase

Campos
 

- \_formato\_hecho : int
- \_formato\_reglas : int
- \_formato\_variables : int
- base\_conocimiento : LecturaBaseConocimiento
- FORMATO\_ID\_MAS CONTENIDO : int
- FORMATO\_SOLO CONTENIDO : int
- FORMATO\_SOLO\_ID : int

Propiedades
 

- formato\_hechos : int
- formato\_reglas : int
- formato\_variables : int
- mostrar\_log\_accion\_consultando\_hecho : bool
- mostrar\_log\_accion\_elegida\_mejor\_regla : bool
- mostrar\_log\_accion\_hecho\_objetivo\_actual : bool
- mostrar\_log\_accion\_hecho\_objetivo\_principal : bool
- mostrar\_log\_accion\_ingresando\_a\_variables\_conocidas : bool
- mostrar\_log\_accion\_ingresando\_hecho\_a\_pila\_objetivos : bool
- mostrar\_log\_accion\_mover\_hecho : bool
- mostrar\_log\_accion\_mover\_regla : bool
- mostrar\_log\_accion\_procesando\_respuesta : bool
- mostrar\_log\_accion\_quitando\_hecho\_de\_pila\_objetivos : bool
- mostrar\_log\_accion\_validando\_regla : bool
- mostrar\_log\_hecho : bool
- mostrar\_log\_info : bool
- mostrar\_log\_info\_analizando\_hechos\_inferidos\_regla : bool
- mostrar\_log\_info\_consultando\_hechos : bool
- mostrar\_log\_info\_consultando\_variables\_de\_inicio : bool
- mostrar\_log\_info\_continuando\_proceso : bool
- mostrar\_log\_info\_descartando\_reglas\_de\_igual\_consecuente...
- mostrar\_log\_info\_problema\_no\_solucionado : bool
- mostrar\_log\_info\_problema\_solucionado : bool
- mostrar\_log\_info\_procesando\_hechos\_asociados : bool
- mostrar\_log\_info\_proceso\_detenido : bool
- mostrar\_log\_info\_regla\_no\_validada : bool
- mostrar\_log\_info\_regla\_validada : bool
- mostrar\_log\_info\_sin\_reglas\_para\_inferir\_hecho : bool
- mostrar\_log\_info\_termino\_de\_inferencia\_reglas\_agotadas : b...
- mostrar\_log\_info\_variable\_conocida : bool
- mostrar\_log\_nivel\_hecho : bool
- mostrar\_log\_regla : bool
- mostrar\_log\_variable : bool
- ruta\_base\_conocimiento : string
- texto\_hechos\_disponibles : string
- texto\_hechos\_falsos : string
- texto\_hechos\_verdaderos : string
- texto\_log\_accion\_consultando\_hechos\_final : string
- texto\_log\_accion\_consultando\_hechos\_inicial : string
- texto\_log\_accion\_elegida\_mejor\_regla\_final : string
- texto\_log\_accion\_elegida\_mejor\_regla\_inicial : string
- texto\_log\_accion\_hecho\_objetivo\_actual\_final : string
- texto\_log\_accion\_hecho\_objetivo\_actual\_inicial : string
- texto\_log\_accion\_hecho\_objetivo\_principal\_final : string
- texto\_log\_accion\_hecho\_objetivo\_principal\_inicial : string
- texto\_log\_accion\_ingresando\_a\_variables\_conocidas\_final : s...
- texto\_log\_accion\_ingresando\_a\_variables\_conocidas\_inicial : ...
- texto\_log\_accion\_ingresando\_hecho\_a\_pila\_objetivos\_final : ...
- texto\_log\_accion\_ingresando\_hecho\_a\_pila\_objetivos\_inicial...
- texto\_log\_accion\_mover\_hecho\_central : string
- texto\_log\_accion\_mover\_hecho\_final : string
- texto\_log\_accion\_mover\_hecho\_inicial : string
- texto\_log\_accion\_mover\_regla\_central : string
- texto\_log\_accion\_mover\_regla\_final : string
- texto\_log\_accion\_mover\_regla\_inicial : string
- texto\_log\_accion\_procesando\_respuesta\_inicial : string
- texto\_log\_accion\_procesando\_respuesta\_medio : string
- texto\_log\_accion\_quitando\_hecho\_de\_pila\_objetivos\_final : s...
- texto\_log\_accion\_quitando\_hecho\_de\_pila\_objetivos\_inicial : ...
- texto\_log\_accion\_validando\_regla\_final : string
- texto\_log\_accion\_validando\_regla\_inicial : string
- texto\_log\_hecho : string
- texto\_log\_info : string
- texto\_log\_info\_analizando\_hechos\_inferidos\_regla : string
- texto\_log\_info\_consultando\_hechos : string
- texto\_log\_info\_consultando\_variables\_de\_inicio : string
- texto\_log\_info\_continuando\_proceso : string
- texto\_log\_info\_descartando\_reglas\_de\_igual\_consecuente : s...
- texto\_log\_info\_problema\_no\_solucionado : string
- texto\_log\_info\_problema\_solucionado : string
- texto\_log\_info\_procesando\_hechos\_asociados : string
- texto\_log\_info\_proceso\_detenido : string
- texto\_log\_info\_regla\_no\_validada : string
- texto\_log\_info\_regla\_validada : string
- texto\_log\_info\_sin\_reglas\_para\_inferir\_hecho\_final : string
- texto\_log\_info\_sin\_reglas\_para\_inferir\_hecho\_inicial : string
- texto\_log\_info\_termino\_de\_inferencia\_reglas\_agotadas : stri...
- texto\_log\_info\_variable\_conocida\_final : string
- texto\_log\_info\_variable\_conocida\_inicial : string
- texto\_log\_nivel\_hecho : string
- texto\_log\_regla : string
- texto\_log\_variable : string
- texto\_reglas\_candidatas : string
- texto\_reglas\_disponibles : string
- texto\_reglas Eliminadas : string

Métodos
 

- Equals() : bool
- formatearAtributo() : string (+ 2 sobrecargas)
- mostrarInfo() : bool
- mostrarInfosPorDefecto() : void
- nombreLista() : string (+ 1 sobrecarga)
- ProcesadorLogInferencia() (+ 1 sobrecarga)
- ProcesarLineaDeLoggeo() : string
- textoLogPorDefecto() : void





**AccesoDatos**  
Clase

**Campos**

- \_existe\_base\_conocimiento : bool
- \_ruta\_carpeta\_archivos : string
- archivo\_configuracion : string
- carpeta\_archivos\_rtf : string
- carpeta\_configuracion : string
- HECHO : int
- REGLA : int
- VARIABLE : int

**Propiedades**

- existe\_base\_conocimiento : bool
- ruta\_archivo\_configuracion : string
- ruta\_carpeta\_archivos : string
- ruta\_carpeta\_configuracion : string
- ruta\_carpeta\_rtf : string

**Métodos**

- AccesoDatos() (+ 1 sobrecarga)
- actualizarHecho() : void
- actualizarMetadatos() : void
- actualizarRegla() : void
- actualizarVariable() : void
- comprobarArchivo() : bool
- comprobarCarpeta() : bool
- eliminarArchivoDeConfiguracion() : void
- eliminarCarpetaBaseConocimiento() : void
- eliminarHecho() : void
- eliminarRegla() : void
- eliminarVariable() : void
- extraerHecho() : Hecho
- extraerMetadatosBaseConocimiento() : MetadatosBaseDeConocimiento
- extraerRegla() : Regla
- extraerVariable() : Variable
- ingresarNuevaRegla() : void
- ingresarNuevaVariable() : void
- ingresarNuevoHecho() : void
- ingresarObjetoMetadatos() : void
- inicializarCarpetaArchivos() : void
- listarArchivosEnDirectorio() : string[]

## 3.4. Sub módulo gestión del conocimiento

**GestoradorBaseConocimiento**  
 Clase

Campos
 

- cantidad\_de\_hechos : int
- cantidad\_de\_reglas : int
- cantidad\_de\_variables : int
- ENCADENAMIENTO\_HACIA\_ADELANTE : int
- ENCADENAMIENTO\_HACIA\_ATRAS : int
- HECHO : int
- manejador\_archivos : AccesoDatos
- REGLA : int
- ultima\_id\_hecho : int
- ultima\_id\_regla : int
- ultima\_id\_variables : int
- VARIABLE : int

Propiedades
 

- existe\_base\_de\_conocimiento : bool
- ruta\_archivo\_configuracion : string
- ruta\_carpeta\_archivos\_rtf : string
- ruta\_carpeta\_base\_conocimiento : string
- ruta\_carpeta\_configuracion : string

Métodos
 

- actualizarAtributosHecho() : void
- actualizarAtributosVariableEnHechos() : void
- actualizarEstadisticas() : void
- agregarElementoListaVariable() : void
- agregarHechoAlAntecedenteDeRegla() : void
- agregarHechoAlConsecuente() : void
- agregarNuevaRegla() : string (+ 2 sobrecargas)
- agregarNuevaVariable() : string
- agregarNuevoHecho() : string (+ 2 sobrecargas)
- buscarHecho() : string (+ 3 sobrecargas)
- comprobarAmbigüedadRecursivaEnReglas() : List<string>
- comprobarBaseDeConocimiento() : List<string>
- comprobarIntegridadReglas() : List<string>
- comprobarIntegridadVariables() : List<string>
- comprobarNombreVariable() : bool
- comprobarReglaExistente() : string (+ 3 sobrecargas)
- comprobarUnicidadReglas() : List<string>
- desmarcarChequeoDeConsistenciaEnHechosYReglas() : void (+ 1 sobrecarga)
- eliminarElementoListaVariable() : void
- eliminarHecho() : void
- eliminarHechoDeAntecedenteDeRegla() : void
- eliminarHechoDeConsecuenteDeRegla() : void
- eliminarHechoDeRegla() : void
- eliminarRegla() : void
- eliminarVariable() : void
- GestoradorBaseConocimiento() (+ 1 sobrecarga)
- iniciarNuevaBaseDeConocimiento() : void
- leerHecho() : Hecho
- leerMetadatos() : MetadatosBaseDeConocimiento
- leerRegla() : Regla
- leerVariable() : Variable
- limpiarHechosNoUtilizadosBaseDeConocimiento() : void
- listarHechos() : string[]
- listarHechosConVariable() : string[] (+ 3 sobrecargas)
- listarReglas() : string[]
- listarReglasConHecho() : string[] (+ 1 sobrecarga)
- listarVariables() : string[]
- metadatosCambiarChequeoBaseConocimiento() : void
- metadatosCambiarEncadenamiento() : void
- metadatosCambiarRutaImageLogo() : void
- metadatosCambiarRutaRtfDescriptivo() : void
- metadatosCambiarTitulo() : void
- metadatosDesmarcarChequeoBaseConocimiento() : void
- modificarAtributosVariableNumerica() : void (+ 1 sobrecarga)
- modificarMetadatosVariable() : bool
- modificarRegla() : string
- modificarRutasArchivosVariable() : bool

Tipos anidados
 

**InfoVariable**  
 Struct





### EncadenamientoHaciaAdelante

Clase

- Campos**
  - \_codigo\_de\_salida\_proceso : int
  - \_encadenamiento\_inicializado : bool
  - base\_conocimiento : AccesoDatos
  - lista\_hechos\_disponibles : List<string>
  - lista\_hechos\_falsos : List<string>
  - lista\_hechos\_verdaderos : List<string>
  - lista\_reglas\_candidatas : List<InfoRegla>
  - lista\_reglas\_disponibles : List<InfoRegla>
  - lista\_reglas\_eliminadas : List<InfoRegla>
  - lista\_variables\_conocidas : List<string>
  - lista\_variables\_de\_inicio : List<string>
  - lista\_variables\_disponibles : List<string>
  - log : List<string>
  - tiempo\_regla : int
- Propiedades**
  - codigo\_de\_salida\_proceso : int
  - encadenamiento\_inicializado : bool
  - loggeo\_inferencia : List<string>
- Métodos**
  - actualizarEvaluacionHecho() : bool
  - actualizarReglasConHechoVerdaderoAntecedente() : void
  - actualizarValidacionDeRegla() : void
  - actualizarValidacionInfoRegla() : void
  - actualizarVariable() : void (+ 2 sobrecargas)
  - actualizarVariableHechoInferido() : void
  - agregarLog() : void
  - aumentarConteoInfoRegla() : void
  - buscarHechosConVariable() : string[]
  - buscarReglasConHechoEnConsecuentes() : string[]
  - descartarReglasConHechoConsecuente() : void
  - elegirMejorReglaCandidata() : string
  - eliminarReglasConHechoFalsoEnElAntecedente() : void
  - EncadenamientoHaciaAdelante() (+ 1 sobrecarga)
  - inferencia() : void
  - inicializacionVariables() : void
  - inicializarEncadenamiento() : string
  - inicializarHechos() : void
  - inicializarReglas() : void
  - moverHecho() : void
  - moverRegla() : void (+ 1 sobrecarga)
  - procesarHecho() : bool
  - procesarRespuestaVariable() : void
  - rescatarInformacionRegla() : InfoRegla
  - variableConocida() : bool
- Eventos**
  - evento\_confimar\_hecho : DelegadoConfirmarHecho
  - evento\_consultar\_variable : DelegadoConsultarVariable
  - evento\_informacion\_inferencia : DelegadoInformacionInferencia
- Tipos anidados**

### EncadenamientoHaciaAtras

Clase

- Campos**
  - \_codigo\_de\_salida\_proceso : int
  - \_encadenamiento\_inicializado : bool
  - \_hecho\_objetivo\_establecido : bool
  - base\_conocimiento : AccesoDatos
  - lista\_hechos\_consecuente\_con\_variables\_objetivo : ArrayList
  - lista\_hechos\_disponibles : List<string>
  - lista\_hechos\_falsos : List<string>
  - lista\_hechos\_verdaderos : List<string>
  - lista\_reglas\_candidatas : List<InfoRegla>
  - lista\_reglas\_disponibles : List<InfoRegla>
  - lista\_reglas\_eliminadas : List<InfoRegla>
  - lista\_variables\_conocidas : List<string>
  - lista\_variables\_disponibles : List<string>
  - lista\_variables\_objetivos : List<string>
  - log : List<string>
  - pila\_hechos\_a\_verificar : Stack<HechoPila>
  - tiempo\_regla : int
- Propiedades**
  - codigo\_de\_salida\_proceso : int
  - encadenamiento\_inicializado : bool
  - hecho\_objetivo\_establecido : bool
  - loggeo\_inferencia : List<string>
- Métodos**
  - actualizarEvaluacionHecho() : bool
  - actualizarReglasConHechoConsecuente() : void
  - actualizarReglasConHechoVerdaderoAntecedente() : void
  - actualizarValidacionInfoRegla() : void
  - actualizarVariableHechoInferido() : void
  - agregarLog() : void
  - aumentarConteoInfoRegla() : void
  - buscarReglasConHechoEnConsecuentes() : string[]
  - compararAfinidadDeHechos() : bool
  - elegirMejorReglaCandidata() : string
  - eliminarReglasConHechoEnElAntecedente() : void
  - EncadenamientoHaciaAtras() (+ 1 sobrecarga)
  - establecerHechoObjetivo() : void
  - inferencia() : void
  - inicializacionVariables() : void
  - inicializarEncadenamiento() : string
  - inicializarHechos() : void
  - inicializarReglas() : void
  - moverHecho() : void
  - moverRegla() : void (+ 1 sobrecarga)
  - obtenerPosiblesVariablesObjetivos() : string[]
  - obtenerPosibleHechosObjetivos() : string[]
  - procesarHecho() : bool
  - procesarRespuestaVariable() : void
  - rescatarInformacionRegla() : InfoRegla
  - variableConocida() : bool
- Eventos**
  - evento\_confimar\_hecho : DelegadoConfirmarHecho
  - evento\_consultar\_variable : DelegadoConsultarVariable
  - evento\_informacion\_inferencia : DelegadoInformacionInferencia
- Tipos anidados**

**LecturaBaseConocimiento**  
Clase

- Campos
  - base\_conocimiento : AccesoDatos
- Propiedades
  - existe\_base\_conocimiento : bool
  - ruta\_base\_conocimiento : string
- Métodos
  - LecturaBaseConocimiento() (+ 1 sobrecarga)
  - leerHecho() : Hecho
  - leerMetadatos() : MetadatosBaseDeConocimiento
  - leerRegla() : Regla
  - leerVariable() : Variable
  - listarHechos() : string[]
  - listarReglas() : string[]
  - listarVariables() : string[]

**ConstantesShell**  
Static Class

- Campos
  - CONTINUAR\_PROCESO : int
  - DETENER\_PROCESO : int
  - HECHO\_CONFIRMADO : int
  - HECHO\_DESCARTADO : int
  - HECHOS\_DISPONIBLES : int
  - HECHOS\_FALSOS : int
  - HECHOS\_VERDADEROS : int
  - INFERENCIA\_DETENIDA\_POR\_EL\_USUARIO : int
  - INFERENCIA\_DETENIDA\_PROBLEMA\_NO\_SOLUCIONADO : int
  - INFERENCIA\_DETENIDA\_PROBLEMA\_SOLUCIONADO : int
  - INFERENCIA\_TERMINADA\_PROBLEMA\_NO\_SOLUCIONADO\_REGLAS\_FALTANTES : int
  - LOG\_ACCION\_CONSULTANDO\_HECHO : int
  - LOG\_ACCION\_ELEGIDA\_MEJOR\_REGLA : int
  - LOG\_ACCION\_HECHO\_OBJETIVO\_ACTUAL : int
  - LOG\_ACCION\_HECHO\_OBJETIVO\_PRINCIPAL : int
  - LOG\_ACCION\_INGRESANDO\_A\_VARIABLES\_CONOCIDAS : int
  - LOG\_ACCION\_INGRESANDO\_HECHO\_A\_PILA\_OBJETIVOS : int
  - LOG\_ACCION\_MOVER\_HECHO : int
  - LOG\_ACCION\_MOVER\_REGLA : int
  - LOG\_ACCION\_PROCESANDO\_RESPUESTA : int
  - LOG\_ACCION\_QUITANDO\_HECHO\_DE\_PILA\_OBJETIVOS : int
  - LOG\_ACCION\_VALIDANDO\_REGLA : int
  - LOG\_HECHO : int
  - LOG\_INFO : int
  - LOG\_INFO\_ANALIZANDO\_HECHOS\_INFERIDOS\_REGLA : int
  - LOG\_INFO\_CONSULTANDO\_HECHOS : int
  - LOG\_INFO\_CONSULTANDO\_VARIABLES\_DE\_INICIO : int
  - LOG\_INFO\_CONTINUANDO\_PROCESO : int
  - LOG\_INFO\_DESCARTANDO\_REGLAS\_DE\_IGUAL\_CONSECUENTE : int
  - LOG\_INFO\_PROBLEMA\_NO\_SOLUCIONADO : int
  - LOG\_INFO\_PROBLEMA\_SOLUCIONADO : int
  - LOG\_INFO\_PROCESANDO\_HECHOS\_ASOCIADOS : int
  - LOG\_INFO\_PROCESO\_DETENIDO : int
  - LOG\_INFO\_REGLA\_NO\_VALIDADA : int
  - LOG\_INFO\_REGLA\_VALIDADA : int
  - LOG\_INFO\_SIN\_REGLAS\_PARA\_INFERIR\_HECHO : int
  - LOG\_INFO\_TERMINO\_DE\_INFERENCIA\_REGLAS\_AGOTADAS : int
  - LOG\_INFO\_VARIABLE\_CONOCIDA : int
  - LOG\_NIVEL\_HECHO : int
  - LOG\_REGLA : int
  - LOG\_VARIABLE : int
  - PROBLEMA\_NO\_SOLUCIONADO : int
  - PROBLEMA\_SOLUCIONADO : int
  - REGLAS\_CANDIDATAS : int
  - REGLAS\_DISPONIBLES : int
  - REGLAS\_ELIMINADAS : int

## 3.6. Diseño de interfaz

### 3.6.1. GestionadorBaseDeConocimiento

#### **actualizarAtributosHecho()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.actualizarAtributosHecho(string id\_hecho)

Método que actualiza el hecho según los cambios de su variable interna

Parámetros

id\_hecho      Id del hecho a actualizar

#### **actualizarEstadisticas()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.actualizarEstadisticas()

Método que actualiza las variables internas del objeto

#### **agregarElementoListaVariable()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarElementoListaVariable(string id\_variable,string elemento\_variable )

Método para agregar un nuevo elemento a lista de una variable TIPO LISTA

Parámetros

id\_variable      ID de la variable a modificar

elemento\_variable      elemento a agregar

#### **agregarHechoAlAntecedenteDeRegla()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarHechoAlAntecedenteDeRegla(string id\_regla,string id\_hecho )

Método que ingresa un nuevo hecho al antecedente de la regla

Parámetros

id\_regla      Id de la regla a modificar

id\_hecho      Id del hecho a ingresar

#### **agregarHechoAlConsecuente()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarHecho  
AlConsecuente(string id\_regla,string id\_hecho )

Método que ingresa un nuevo hecho al consecuente de la regla

Parámetros

id\_regla            Id de la regla a modificar

id\_hecho           Id del hecho a ingresar

### **agregarNuevaRegla()** [1/3]

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNueva  
Regla()

Método que agrega una nueva regla vacía a la base de conocimiento

Retorna Id de la regla creada

### **agregarNuevaRegla()** [2/3]

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNueva  
Regla(ArrayList antecedentes,ArrayList consecuente )

Método que ingresa una nueva regla creando los hechos correspondientes

Parámetros

antecedentes    Se debe completar según corresponda la

variableArrayList[ArrayList[id\_variable string ,codicion string ],ArrayList[id\_variable string  
,condicion string ,valor double],ArrayList[id\_variable string ,codicion string ,valor string] ]

consecuente    Se debe completar sólo un arraylist correspondiente a la variable  
consecuente

Retorna null si la regla ya se encuentra en la base de conocimiento

### **agregarNuevaRegla()** [3/3]

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNueva  
Regla(string [] ids\_hechos\_antecedente,string id\_hecho\_consecuente )

Método que ingresa una nueva regla creando los hechos correspondientes

Parámetros

ids\_hechos\_antecedente    Identificador de los hechos antecedentes

id\_hecho\_consecuente      Identificador de los hechos consecuentes

Retorna null si la regla ya se encuentra en la base de conocimiento

### **agregarNuevaVariable()**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNueva  
Variable(string nombre\_variable,int tipo\_de\_variable,string [] lista\_de\_opciones = null )

Método para agregar una nueva variable a la base de conocimiento

Parámetros

---

nombre\_variable      Nombre de la variable a ingresar  
tipo\_de\_variable      Tipo de la variable a ingresar según constantes  
BOOLEANO,NUMERICO,LISTA  
lista\_de\_opciones      En caso de ser de tipo LISTA, opciones a guardar en la variable  
Retorna Id de la variable si se agrego correctamente la variable, NULL si el nombre ya  
existe

**agregarNuevoHecho() [1/3]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNuevo  
Hecho(string id\_variable,string condicion )

Agrega un nuevo hecho de tipo BOLEANO a la base de conocimiento

Parámetros

id\_variable      Id de la variable a ingresar en el hecho

condicion      Opción a cotejar según CONSTANTE OPCIONES\_BOLEANO  
{VERDADERO , FALSO}

Retorna Id del hecho creado

**agregarNuevoHecho() [2/3]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNuevo  
Hecho(string id\_variable,string condicion,double valor )

Agrega un nuevo hecho de tipo NUMERICO a la base de conocimiento

Parámetros

id\_variable      Id de la variable a ingresar en el hecho

condición      Opción de de la condición según constante OPCIONES\_NUMERICO  
{ "MENOR", "MENOR O IGUAL", "IGUAL", "MAYOR O IGUAL", "MAYOR" }

valor      Valor a comparar según corresponda la condición impuesta

Retorna Id del hecho creado

**agregarNuevoHecho() [3/3]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.agregarNuevo  
Hecho(string id\_variable,string condicion,string valor\_lista\_variable )

Agrega un nuevo hecho de tipo LISTA a la base de conocimiento

Parámetros

id\_variable      Id de la variable a ingresar en el hecho

condición      Opción de la condición según constante OPCIONES\_LISTA {"ES", "NO ES"}

valor\_lista\_variable      Valor a comparar según corresponda la condición impuesta

Retorna Id del hecho creado

**buscarHecho() [1/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.buscarHecho(Hecho hecho\_buscado)

Método que busca un Hecho por comparación de atributos

Parámetros

hecho\_buscado      Hecho a buscar

Retorna Id del hecho encontrado, null si no se encuentra

#### **buscarHecho() [2/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.buscarHecho(string id\_variable,string condicion )

Método que busca un hecho según su variable y condición

Parámetros

id\_variable      Identificador variable de tipo BOOOLEANO

condición      Condición de la variable en el hecho

Retorna Id del hecho encontrado, null si no se encuentra

#### **buscarHecho() [3/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.buscarHecho(string id\_variable,string condicion,double valor )

Método que busca un hecho según su variable, condición y valor

Parámetros

id\_variable      Identificador variable de tipo NUMERICO

condición      Condición de la variable en el hecho

valor      valor de la variable en la condición

Retorna Id del hecho encontrado, null si no se encuentra

#### **buscarHecho() [4/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.buscarHecho(string id\_variable,string condicion,string valor\_lista\_variable )

Método que busca un hecho según su variable, condición y valor

Parámetros

id\_variable      Identificador variable de tipo LISTA

condición      Condición de la variable en el hecho

valor lista variable      valor de la variable en la condición

Retorna Id del hecho encontrado, null si no se encuentra

#### **comprobarBaseDeConocimiento()**

List<string>

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarBaseDeConocimiento()

**comprobarNombreVariable()**

bool

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarNombreVariable(string nombre\_variable)

Método que comprueba en todas las variables de la base de conocimiento si el nombre existe

Parámetros

nombre\_variable      Nombre a comprobar

Retorna

**comprobarReglaExistente() [1/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarReglaExistente(Regla regla\_buscada,string id\_omitir\_regla = null )

Método que comprueba la existencia de una regla en la base de conocimiento

Parámetros

regla\_buscada      Regla a buscar

id\_omitir\_regla      OPCIONAL      id de una regla a omitir

Retorna id de la regla encontrada, null si hay una regla igual en la base de conocimiento

**comprobarReglaExistente() [2/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarReglaExistente(Hecho [] hechos\_antecedente,Hecho hecho\_consecuente,string id\_omitir\_regla = null )

Método que comprueba la existencia de una regla en la base de conocimiento

Parámetros

hechos\_antecedente      Lista de hechos en el antecedente

hecho\_consecuente      Lista de hechos en el consecuente

id\_omitir\_regla      OPCIONAL      id de una regla a omitir

Retorna id de la regla encontrada, null si hay una regla igual en la base de conocimiento

**comprobarReglaExistente() [3/4]**

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarReglaExistente(ArrayList antecedentes,ArrayList consecuente,string id\_omitir\_regla = null )

Método que comprueba la existencia de una regla en la base de conocimiento

Parámetros

antecedentes      Se debe completar según corresponda la

variableArrayList[ArrayList[id\_variable string ,codicion string ],ArrayList[id\_variable string ,condicion string ,valor double],ArrayList[id\_variable string ,codicion string ,valor string] ]

consecuente      Se debe completar sólo un arraylist correspondiente a la variable

consecuente

id\_omitir\_regla      OPCIONAL      id de una regla a omitir

---

Retorna null si la regla ya se encuentra en la base de conocimiento

**comprobarReglaExistente()** [4/4]

string

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.comprobarReglaExistente(string [] ids\_hechos\_antecedente,string id\_hecho\_consecuente,string id\_omitir\_regla = null )

Método que comprueba la existencia de una regla en la base de conocimiento

Parámetros

ids\_hechos\_antecedente      Ids de los hechos antecedentes

id\_hecho\_consecuente      Id del hecho consecuente

id\_omitir\_regla      OPCIONAL      id de una regla a omitir

Retorna Id de la regla encontrada, null si la regla no se ha encontrado

**desmarcarChequeoDeConsistenciaEnHechosYReglas()** [1/2]

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.desmarcarChequeoDeConsistenciaEnHechosYReglas(string id\_variable,bool eliminar\_hecho = false,string valor\_lista\_especifico = null )

Método que busca los hechos y reglas en que influye la variable y cambia su estado de chequeo

Parámetros

id\_variable      ID de la variable a buscar

eliminar\_hecho      Indica si los hechos encontrados deben ser eliminados en el proceso

**desmarcarChequeoDeConsistenciaEnHechosYReglas()** [2/2]

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.desmarcarChequeoDeConsistenciaEnHechosYReglas(string id\_variable,double rango\_min,double rango\_max )

Método que busca los hechos y reglas en que influye la variable y cambia su estado de chequeo

Parámetros

id\_variable      ID de la variable de tipo NUMERICO

rango\_min      Especifica el valor mínimo que puede tener la condición

rango\_max      Especifica el valor máximo que puede tener la condición

**eliminarElementoListaVariable()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarElementoListaVariable(string id\_variable,string elemento\_variable )

Método para eliminar un elemento de la lista de una Variable TIPO LISTA

Parámetros

id\_variable      ID de la variable a modificar



---

elemento\_variable      Elemento a eliminar de la variable

**eliminarHechoDeAntecedenteDeRegla()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarHechoDeAntecedenteDeRegla(string id\_regla,string id\_hecho )

Método para eliminar un hecho del antecedente de una regla

Parámetros

id\_regla              Id de la regla a modificar

id\_hecho             Id del hecho a eliminar de la regla

**eliminarHechoDeConsecuenteDeRegla()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarHechoDeConsecuenteDeRegla(string id\_regla)

Método para eliminar el consecuente de una regla

Parámetros

id\_regla              Id de la regla a modificar

**eliminarHechoDeRegla()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarHechoDeRegla(string id\_hecho,string id\_regla )

Método que elimina un hecho específico de una regla

Parámetros

id\_hecho             Id del hecho a eliminar de la regla

id\_regla              Id de la regla a modificar

**eliminarRegla()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarRegla(string id\_regla)

Método para eliminar una regla de la base de conocimiento

Parámetros

id\_regla              Id de la regla a eliminar

**eliminarVariable()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.eliminarVariable(string id\_variable)

Método para eliminar una variable de la base de conocimiento,

Parámetros

id\_variable          ID de la variable a eliminar

---

**iniciarNuevaBaseDeConocimiento()**

void

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.iniciarNuevaBaseDeConocimiento()`

Método para crear las carpetas y el archivo de configuración para una nueva base de conocimientos

**leerHecho()**

Hecho

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.leerHecho(string id_hecho)`

Método para leer un hecho de la base de conocimiento

Parámetros

`id_hecho`      Id del hecho a extraer

Retorna Objeto Hecho encontrado, NULL si no encuentra en la base de conocimientos

**leerMetadatos()**

MetadatosBaseDeConocimiento

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.leerMetadatos()`

Método para leer los metadatos de la base de conocimientos

Retorna Objeto metadatos| null si no existe

**leerRegla()**

Regla

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.leerRegla(string id_regla)`

Método para leer una regla de la base de conocimiento

Parámetros

`id_regla`      Id de la regla a extraer

Retorna Objeto Regla encontrado, NULL si no encuentra en la base de conocimiento

**leerVariable()**

Variable

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.leerVariable(string id_variable)`

Método para leer una Variable de la base de conocimiento

Parámetros

`id_variable`      ID de la variable a leer

Retorna Objeto con la variable encontrada, NULL si no encuentra la variable

**listarHechos()**

string []

`SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarHechos()`

---

Método que devuelve las id de los hechos existentes en el repositorio de datos

Retorna Id de los hechos

### **listarHechosConVariable()** [1/3]

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarHechosConVariable(string id\_variable)

Método que lista todos los hechos que contenga la variable

Parámetros

id\_variable      Id de la variable a buscar

Retorna lista de ID de los hechos que contienen la variable especificada

### **listarHechosConVariable()** [2/3]

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarHechosConVariable(string id\_variable,string valor\_lista\_especifico )

Método que lista todos los hechos que contenga la variable y la valor lista especificado

Parámetros

id\_variable      Id de la variable de tipo LISTA a buscar

valor\_lista\_especifico valor específico de la variable

Retorna lista de ID de los hechos que contienen la variable especificada

### **listarHechosConVariable()** [3/3]

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarHechosConVariable(string id\_variable,double rango\_min,double rango\_max )

Método que lista todos los hechos que contenga la variable de tipo NUMERICA con un valor fuera del rango especificado

Parámetros

id\_variable      ID de la variable de tipo NUMERICA

rango\_min      Especifica el valor mínimo que puede tener la condición

rango\_max      Especifica el valor máximo que puede tener la condición

Retorna lista de ID de los hechos que contienen la variable especificada

### **listarReglas()**

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarReglas()

Método que devuelve las id de las reglas existentes en el repositorio de datos

Retorna Id de las reglas

### **listarReglasConHecho()** [1/2]

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarReglasConHecho(string id\_hecho)

---

Método que devuelve todas las reglas que contienen el hecho

Parámetros

id\_hecho      Id del hecho a buscar

Retorna Arreglo con las Id de las reglas que contienen el hecho especificado

### **listarReglasConHecho()** [2/2]

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarReglasConHecho(string [] id\_hechos)

Método que devuelve todas las reglas que contienen el hecho

Parámetros

id\_hechos      Lista de ID de los hechos a buscar

Retorna Arreglo con las Id de las reglas que contienen el hecho especificado

### **listarVariables()**

string []

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.listarVariables()

Método que devuelve las id de las variables existentes en el repositorio de datos

Retorna Id de las variables

### **metadatosCambiarEncadenamiento()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.metadatosCambiarEncadenamiento(int tipo\_de\_encadenamiento)

Método para cambiar el tipo de encadenamiento de la base de conocimiento

Parámetros

tipo\_de\_encadenamiento

### **metadatosCambiarRutaImageLogo()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.metadatosCambiarRutaImageLogo(string ruta\_imagen)

Método para cambiar la ruta de la imagen logo de la base de conocimiento

Parámetros

ruta\_imagen    ruta imagen

### **metadatosCambiarRutaRtfDescriptivo()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.metadatosCambiarRutaRtfDescriptivo(string ruta\_rtf)

Método para cambiar al ruta del rtf descriptivo de la base de conocimiento

Parámetros

ruta\_rtf path del rtf

### **metadatosCambiarTitulo()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.metadatosCambiarTitulo(string titulo)

Método para cambiar el titulo de la base de conocimiento

Parámetros

titulo

### **metadatosDesmarcarChequeoBaseConocimiento()**

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.metadatosDesmarcarChequeoBaseConocimiento()

Método para desmarcar el chequeo de la base de conocimiento

### **modificarAtributosVariableNumerica()** [1/2]

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.modificarAtributosVariableNumerica(string id\_variable, bool cardinal )

Método que modifica los atributos de una variable de tipo NUMERICO

Parámetros

cardinal          indica si los valores de la variable serán de tipo Real o Cardinal

### **modificarAtributosVariableNumerica()** [2/2]

void

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.modificarAtributosVariableNumerica(string id\_variable, bool rango\_limitado, double rango\_min = -99999999, double rango\_max = 99999999 )

Método que modifica los atributos de una variable de tipo NUMERICO

Parámetros

id\_variable      id de la variable a modificar

rango\_limitado      establece si la variable debera estar en un rango definido

rango\_min      Establece el valor mínimo que puede tomar la variable

rango\_max      Establece el valor máximo que puede tomar la variable

### **modificarMetadatosVariable()**

bool

SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.modificarMetadatosVariable(string id\_variable, bool variable\_de\_inicio, bool variable\_preguntable\_al\_usuario, bool variable\_objetivo, string nombre\_variable = null, string texto\_consulta = null, string ruta\_texto\_descriptivo = null, string ruta\_imagen\_descriptiva = null )

Método para modificar los metadatos de un Objeto Variable

Parámetros

id\_variable      identificador de la variable  
variable\_de\_inicio      Establece si la variable será preguntada al inicio de la inferencia en el encadenamiento hacia adelante  
variable\_preguntable\_al\_usuario      Establece si la variable puede ser preguntada al usuario  
variable\_objetivo      Establece si la variable es objetivo del encadenamiento hacia atrás  
nombre\_variable      modifica en nombre de la variable  
texto\_consulta      Texto con el cual se va a mostrar la variable  
ruta\_texto\_descriptivo      ruta del archivo con la descripción de la variable  
ruta\_imagen\_descriptiva      ruta de la imagen descriptiva de la variable  
Retorna

### **modificarRegla()**

string  
SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.modificarRegla (string id\_regla, ArrayList antecedentes, ArrayList consecuente )  
Método que modifica una regla existente en la base de conocimiento  
Parámetros  
id\_regla      Identificador de la regla  
antecedentes      Se debe completar según corresponda la  
variable ArrayList [ ArrayList [ id\_variable string , codicion string ], ArrayList [ id\_variable string , condicion string , valor double ], ArrayList [ id\_variable string , codicion string , valor string ] ]  
consecuente      Se debe completar sólo un arraylist correspondiente a la variable consecuente  
Retorna null si actualizo correctamente, id de la regla existente en caso contrario

### **modificarRutasArchivosVariable()**

bool  
SistemaExpertoLib.GestionDelConocimiento.GestionadorBaseConocimiento.modificarRutas ArchivosVariable (string id\_variable, string ruta\_texto\_descriptivo = null, string ruta\_imagen\_descriptiva = null )  
Método para modificar las rutas de la Variable  
Parámetros  
id\_variable      id de la variable a modificar  
ruta\_texto\_descriptivo      Ruta texto descriptivo de la variable  
ruta\_imagen\_descriptiva      Ruta imagen descriptiva de la variable  
Retorna Member Dat

### 3.6.2. Encadenamiento hacia adelante

#### **inferencia()**

void SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.inferencia()

Método que realiza la inferencia del sistema experto

#### **inicializarEncadenamiento()**

string

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.inicializarEncadenamiento()

Método que inicializa la base de conocimiento y los atributos del encadenamiento para su inferencia

Retorna

Un texto con los errores encontrados|null si esta todo correcto

#### **codigo\_de\_salida\_proceso**

int

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.codigo\_de\_salida\_proceso

get

Obtiene la forma de termino del proceso de inferencia

#### **encadenamiento\_inicializado**

bool

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.encadenamiento\_inicializado

get

Variable que indica si el objeto ha sido inicializado

#### **log\_inferencia**

List<string>

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.loggeo\_inferencia

get

Lista ordenada con el log de la inferencia

#### **evento\_confimar\_hecho**

DelegadoConfirmarHecho

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.evento\_confimar\_hecho

Evento para confirmar hecho según constantes de control se retorna un int con las opciones segun constates

[HECHO\_VALIDADO,PROBLEMA\_SOLUCIONADO,CONTINUAR\_PROCESO]

#### **evento\_consultar\_variable**

DelegadoConsultarVariable

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.evento\_consultar\_variable

Evento de consulta variable Se debe retornar un ArrayList de la siguiente forma

BOOLEANO {(string)id\_,variable,(bool) valor} NUEMRICO {(string)id\_,variable,(double) valor} LISTA {(string)id\_,variable,(string) opcion}

#### **evento\_informacion\_inferencia**

DelegadoInformacionInferencia

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAdelante.evento\_informacion\_inferencia

Evento que muestra información al usuario acerca de la inferencia

### 3.6.3. Encadenamiento hacia atrás

#### **EncadenamientoHaciaAtras()** [1/2]

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.EncadenamientoHaciaAtras()

Constructor

#### **EncadenamientoHaciaAtras()** [2/2]

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.EncadenamientoHaciaAtras(string ruta\_base\_conocimiento)

Constructor

#### **establecerHechoObjetivo()**

void

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.establecerHechoObjetivo(string id\_hecho)

Método que establece el hecho objetivo del encadenamiento

parámetros

id\_hecho      Id del hecho objetivo, debe pertenecer a la lista de hechos objetivos posibles inferencia()

void SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.inferencia()

Método que realiza la inferencia del sistema experto



---

**inicializarEncadenamiento()**

string

`SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.inicializarEncadenamiento()`

Método que inicializa la base de conocimiento y los atributos del encadenamiento para su inferencia

Retorna

Un texto con los errores encontrados|null si esta todo correcto

**obtenerPosiblesVariablesObjetivos()**

string []

`SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.obtenerPosiblesVariablesObjetivos()`

Método que devuelve las posibles variables objetivos para la inferencia.

Retorna

Ids Variables objetivos, null si no existen

**obtenerPosibleHechosObjetivos()**

string []

`SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.obtenerPosibleHechosObjetivos(string id_variable)`

Método que devuelve los posibles hechos objetivo en funcion de la variable escogida

parámetros

id\_variable      Id variable, perteneciente a las posibles variables objetivos

Retorna

Lista de IDs hechos objetivos asociados a la variable| null si no existe

**codigo\_de\_salida\_proceso**

int

`SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.codigo_de_salida_proceso`

get

Obtiene la forma de termino del proceso de inferencia

**encadenamiento\_inicializado**

bool

`SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.encadenamiento_inicializado`

get

Variable que indica si el objeto ha sido inicializado

---

**hecho\_objetivo\_establecido**

bool

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.hecho\_objetivo\_establecido

get

Indica si el hecho objetivo ha sido establecido para la inferencia

**log\_inferencia**

List&lt;string&gt;

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.loggeo\_inferencia

get

Lista ordenada con el log de la inferencia

**evento\_confimar\_hecho**

DelegadoConfirmarHecho

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.evento\_confimar\_hecho

Evento para confirmar hecho según constantes de control se retorna un int con las opciones según constantes

[HECHO\_VALIDADO, PROBLEMA\_SOLUCIONADO, CONTINUAR\_PROCESO]

**evento\_consultar\_variable**

DelegadoConsultarVariable

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.evento\_consultar\_variable

Evento de consulta variable Se debe retornar un ArrayList de la siguiente forma

BOOLEANO {(string)id\_,variable,(bool) valor} NUMERICO {(string)id\_,variable,(double) valor} LISTA {(string)id\_,variable,(string) opcion}

**evento\_informacion\_inferencia**

DelegadoInformacionInferencia

SistemaExpertoLib.MotorDeInferencia.EncadenamientoHaciaAtras.evento\_informacion\_inferencia

Evento que muestra información al usuario acerca de la inferencia

### 3.6.4. Lectura de la base de conocimiento

**LecturaBaseConocimiento()** [1/2]

SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.LecturaBaseConocimiento()

**LecturaBaseConocimiento()** [2/2]

SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.LecturaBaseConocimiento(string ruta\_carpeta\_archivos)

---

## Constructor

### Parámetros

ruta\_carpeta\_archivos          Ruta del repositorio de la base de conocimiento

Member Function Documentation

### **leerHecho()**

Hecho SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.leerHecho(string id\_hecho)

Método para leer un hecho de la base de conocimiento

### Parámetros

id\_hecho          Id del hecho requerida

retorna

Objeto tipo hecho | null si no existe

### **leerMetadatos()**

MetadatosBaseDeConocimiento

SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.leerMetadatos()

Método para extraer los metadatos correspondiente a la base de conocimiento

retorna

Objeto MetadatosBaseDeConocimiento | null si no existe

### **leerRegla()**

Regla SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.leerRegla(string id\_regla)

Método para leer un regla de la base de conocimiento

### Parámetros

id\_regla          Id del hecho requerida

retorna

Objeto tipo hecho | null si no existe

### **leerVariable()**

Variable

SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.leerVariable(string id\_variable)

Método para leer una variable de la base de conocimiento

### Parámetros

id\_variable          Id de la variable requerida

retorna

Objeto tipo Variable| null si no existe

**listarHechos()**

string [] SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.listarHechos()

Método que lista todos los hechos existentes en la base de conocimiento

retorna

ids hechos

**listarReglas()**

string [] SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.listarReglas()

Método que lista todas las reglas existentes en la base de conocimiento

retorna

ids regla

**listarVariables()**

string [] SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.listarVariables()

Método que lista todas las variables existentes en la base de conocimiento

retorna

ids de las variables

**existe\_base\_conocimiento**

bool

SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.existe\_base\_conocimiento

get

**ruta\_base\_conocimiento**

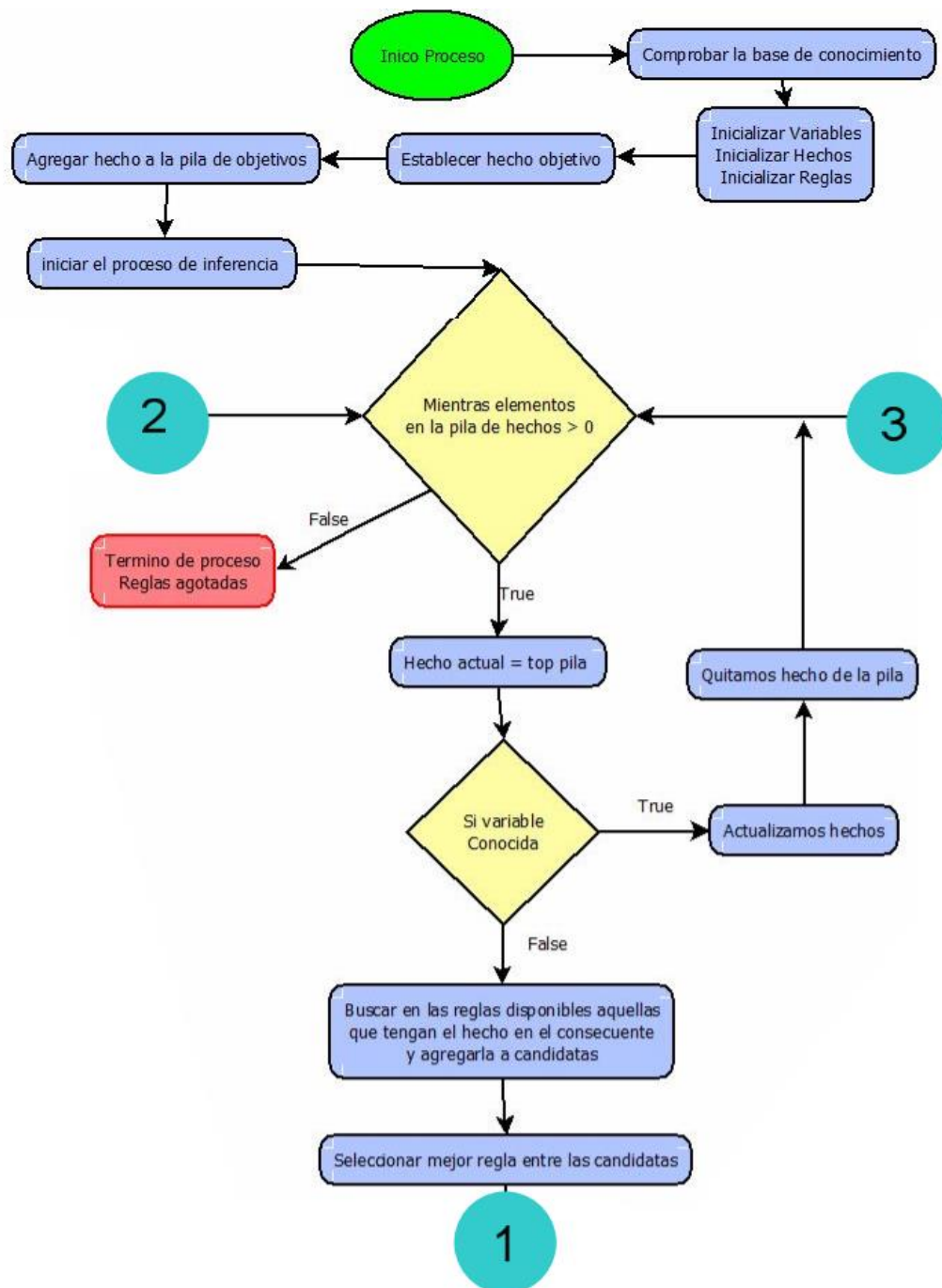
string

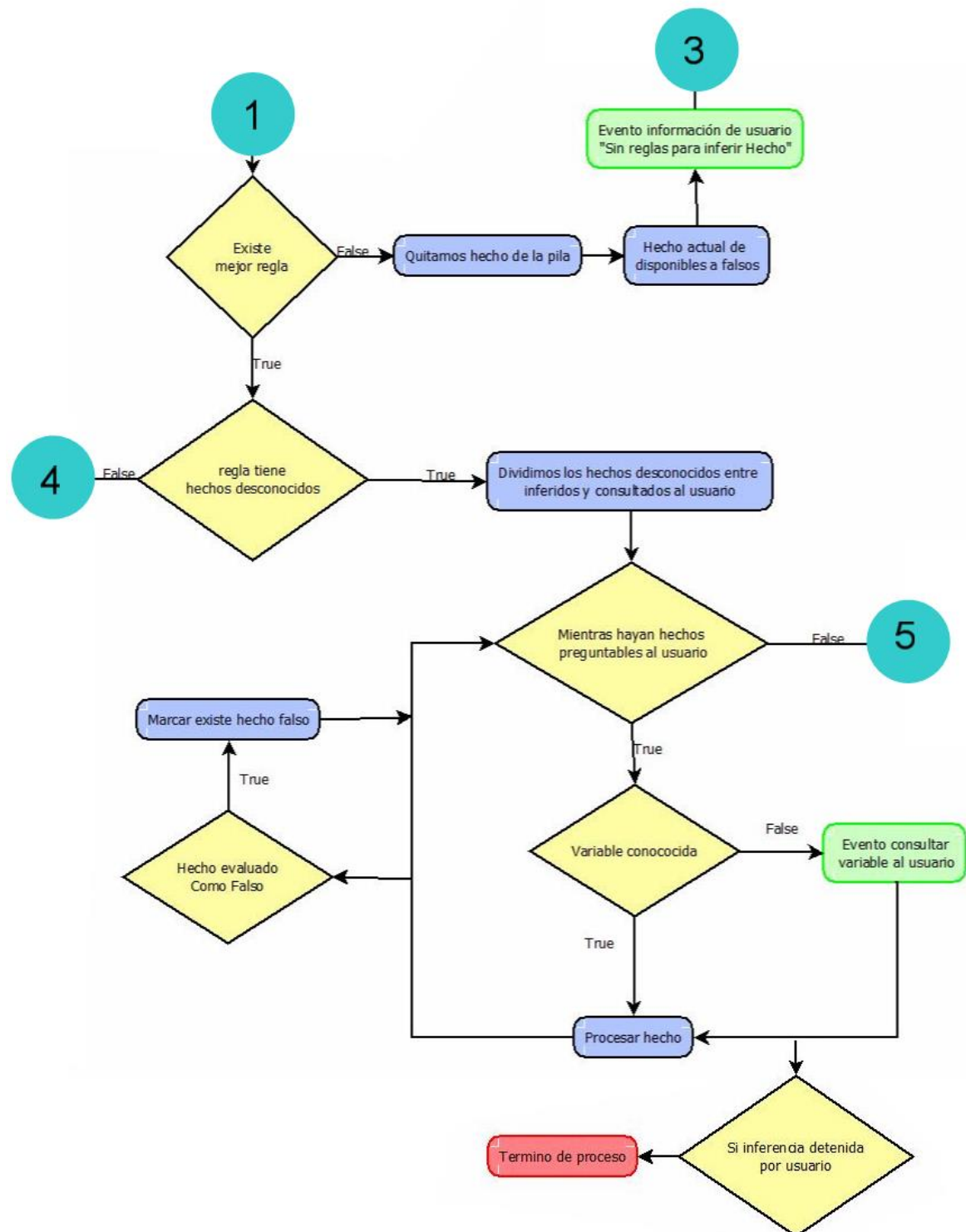
SistemaExpertoLib.MotorDeInferencia.LecturaBaseConocimiento.ruta\_base\_conocimiento

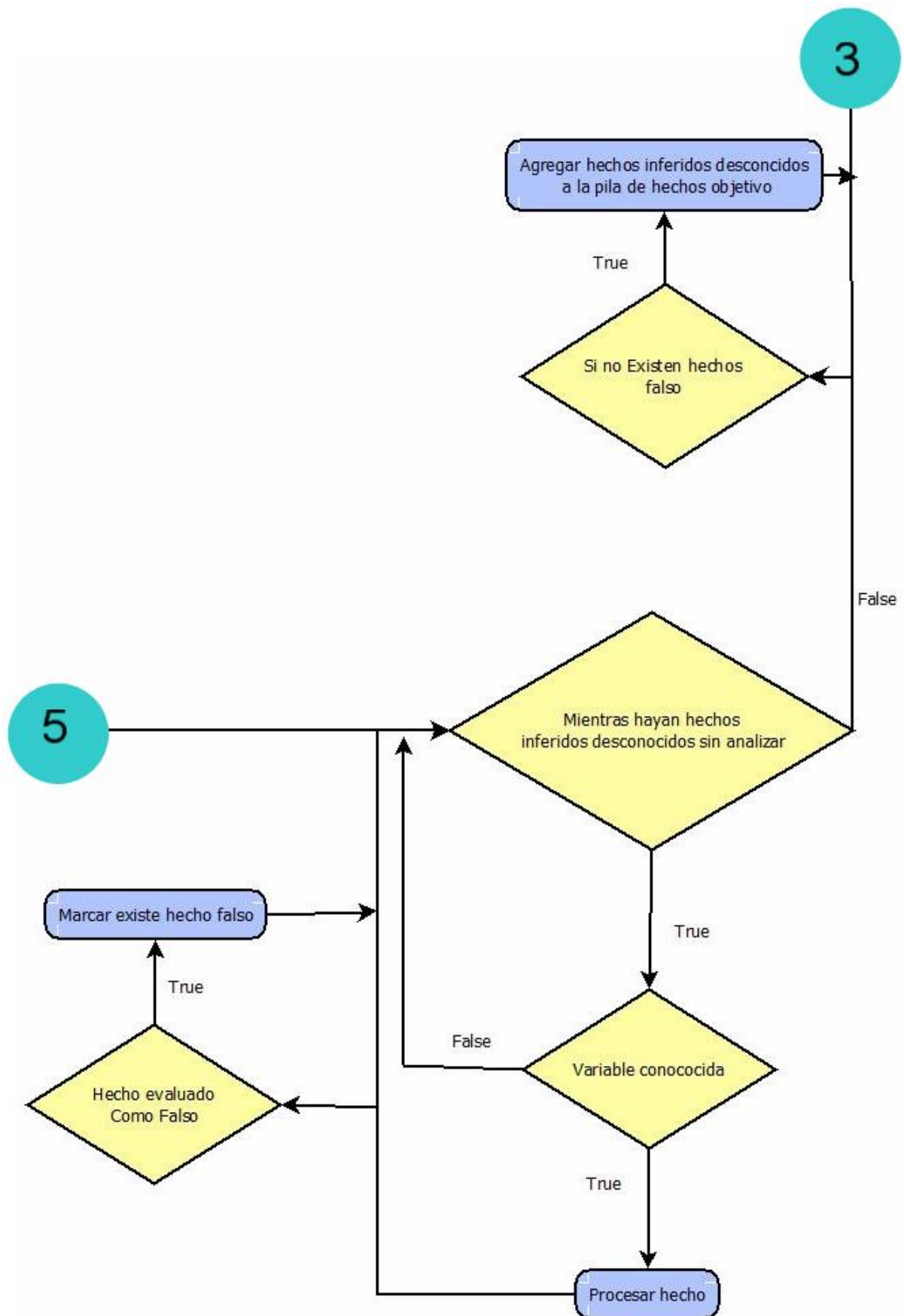


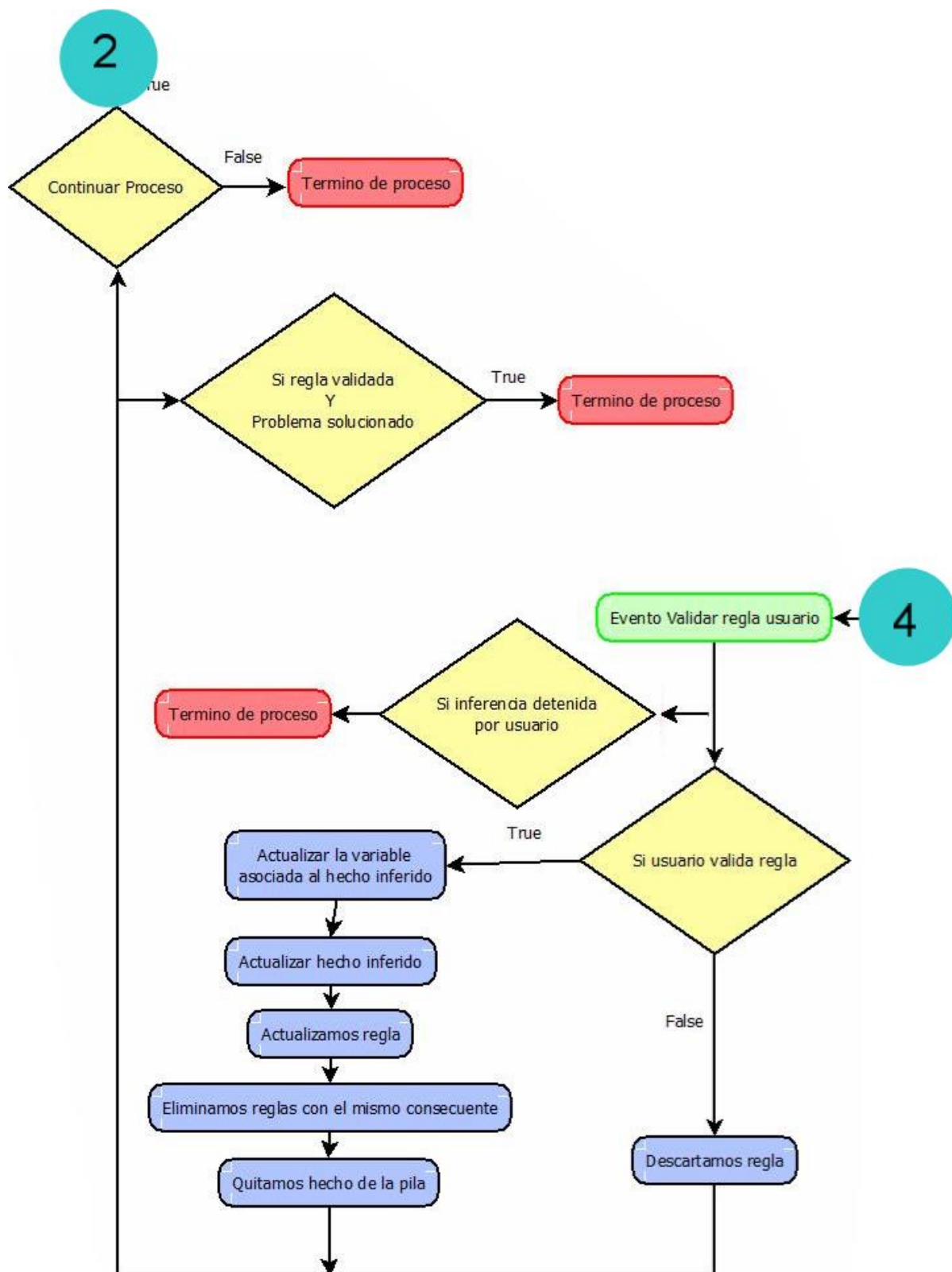
## 3.7. Diseño procedimental

### 3.7.1. Encadenamiento hacia atrás













Las metareglas asociadas al proceso de selección de la mejor regla son las siguientes:

- ★ Criterio de refractariedad: Una regla no puede volver a ser ejecutada después de su validación
- ★ Reglas de igual consecuente: Si una regla es validada, no se ejecutará ninguna regla que contengan el mismo consecuente
- ★ Antecedente falso en las reglas: No se ejecutará ninguna regla que tenga un antecedente falso.
- ★ Consecuente falso en las reglas: No se ejecutará ninguna regla que tenga un consecuente falso.
- ★ Resolución de conflictos: Cada regla lleva asociado la siguiente información
  - tiempo\_regla;
  - numero\_de\_antecedentes;
  - antecedentes\_preguntables\_al\_usuario;
  - antecedentes\_preguntables\_al\_usuario\_conocidos;
  - antecedentes\_inferidos;
  - antecedentes\_inferidos\_conocidos;

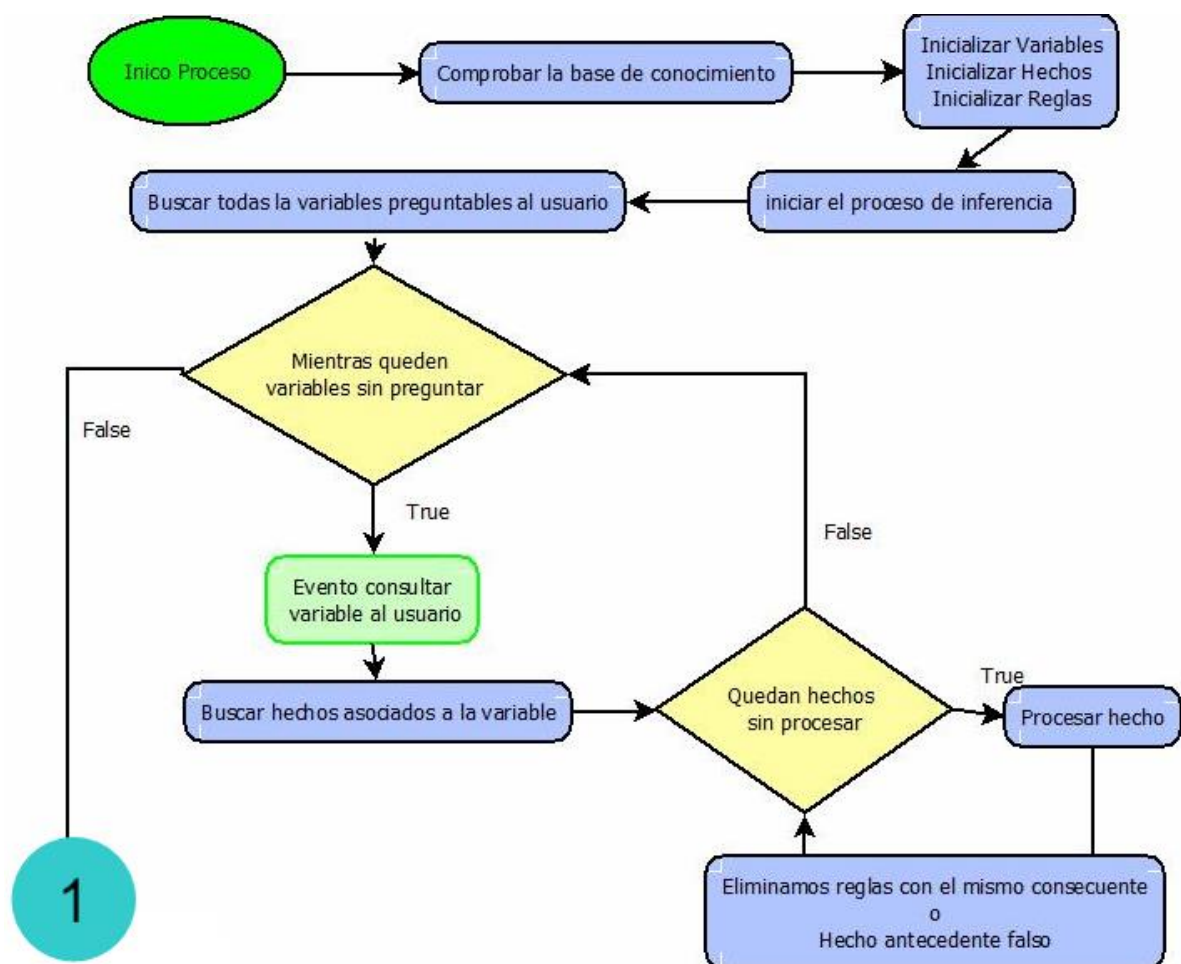
el puntaje de una regla va a estar dado por el numero\_de\_antecedentes menos la suma de antecedentes\_preguntables\_al\_usuario\_conocidos con antecedentes\_inferidos\_conocidos.

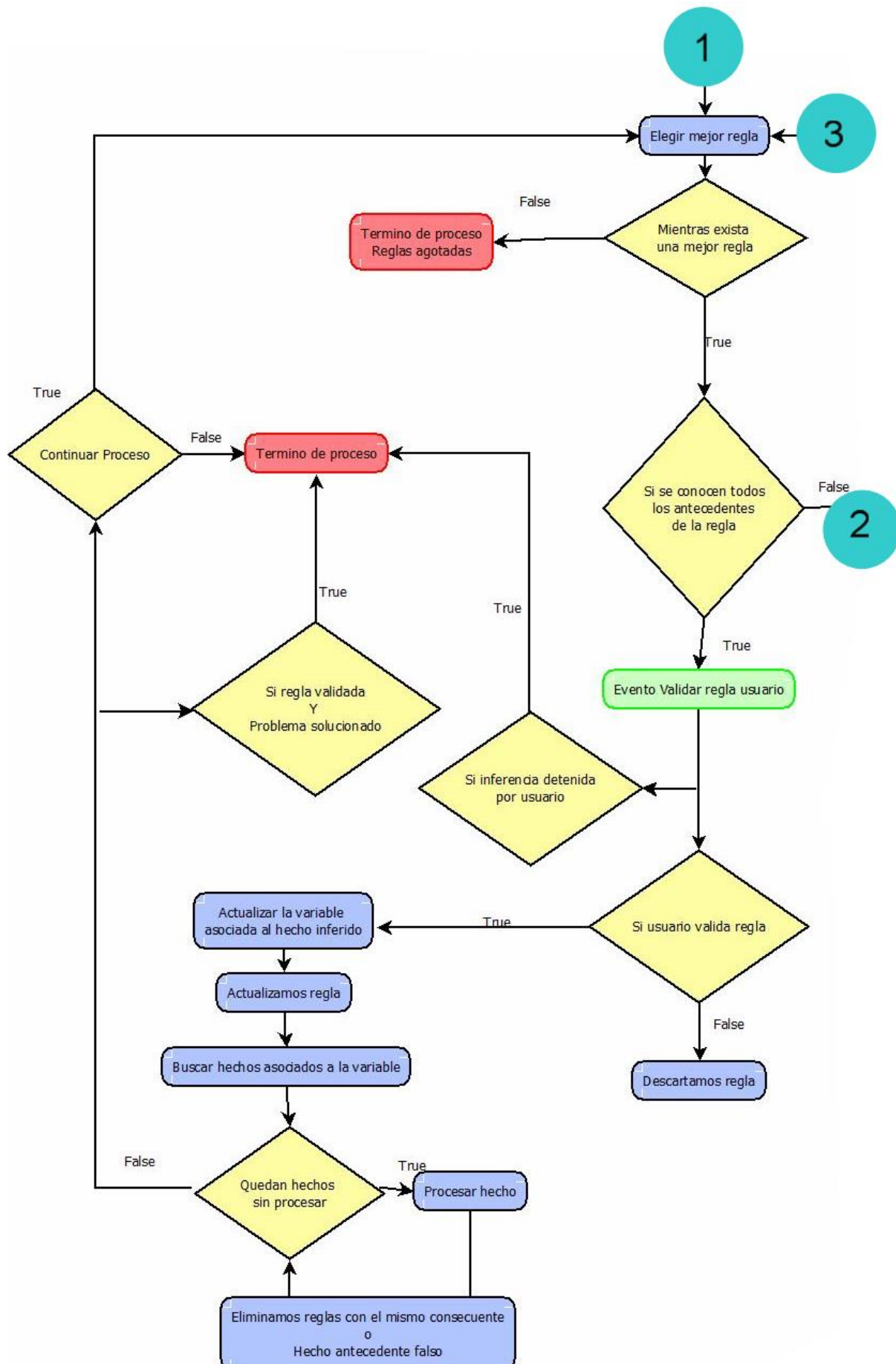
al momento de necesitar resolver el conflicto entre dos reglas se prioriza en el siguiente orden:

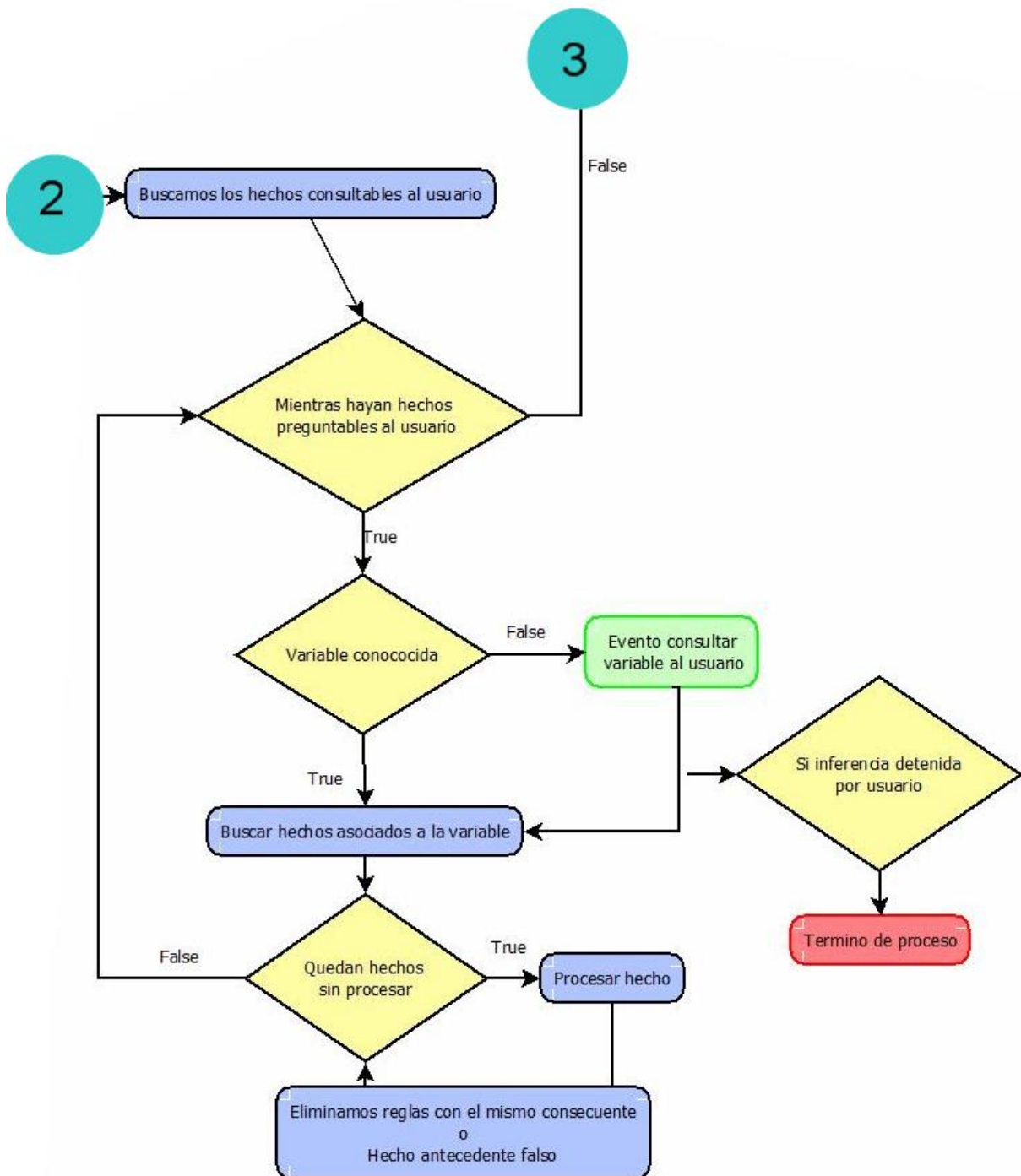
1. Regla con menor puntaje
2. Regla con mayor número de antecedentes preguntables al usuario
3. Regla más actual

## Diseño procedimental

### 3.7.2. Encadenamiento hacia adelante







Las metareglas asociadas al proceso de selección de la mejor regla son las siguientes:

- ★ Criterio de refractariedad: Una regla no puede volver a ser ejecutada después de su validación
- ★ Reglas de igual consecuente: Si una regla es validada, no se ejecutara ninguna regla que contengan el mismo consecuente
- ★ Antecedente falso en las reglas: No se ejecutara ninguna regla que tenga un antecedente falso.
- ★ Consecuente falso en las reglas: No se ejecutara ninguna regla que tenga un consecuente falso.
- ★ Resolución de conflictos: Cada regla lleva asociado la siguiente información
  - tiempo\_regla;
  - numero\_de\_antecedentes;
  - antecedentes\_preguntables\_al\_usuario;
  - antecedentes\_preguntables\_al\_usuario\_conocidos;
  - antecedentes\_inferidos;
  - antecedentes\_inferidos\_conocidos;

el porcentaje de variables conocidas de una regla va a estar dado por el  $\frac{\text{antecedentes\_preguntables\_al\_usuario\_conocidos} + \text{antecedentes\_inferidos\_conocidos}}{\text{numero\_de\_antecedentes}}$  mas el de  $\frac{\text{antecedentes\_inferidos\_conocidos}}{\text{numero\_de\_antecedentes}}$  al momento de necesitar resolver el conflicto entre dos reglas se prioriza en el siguiente orden:

1. Regla con mayor porcentaje de variables conocidas
2. Regla con mayor número de antecedentes conocidos
3. Regla más actual

## ***4. Módulo: Aplicación Tot IDE***

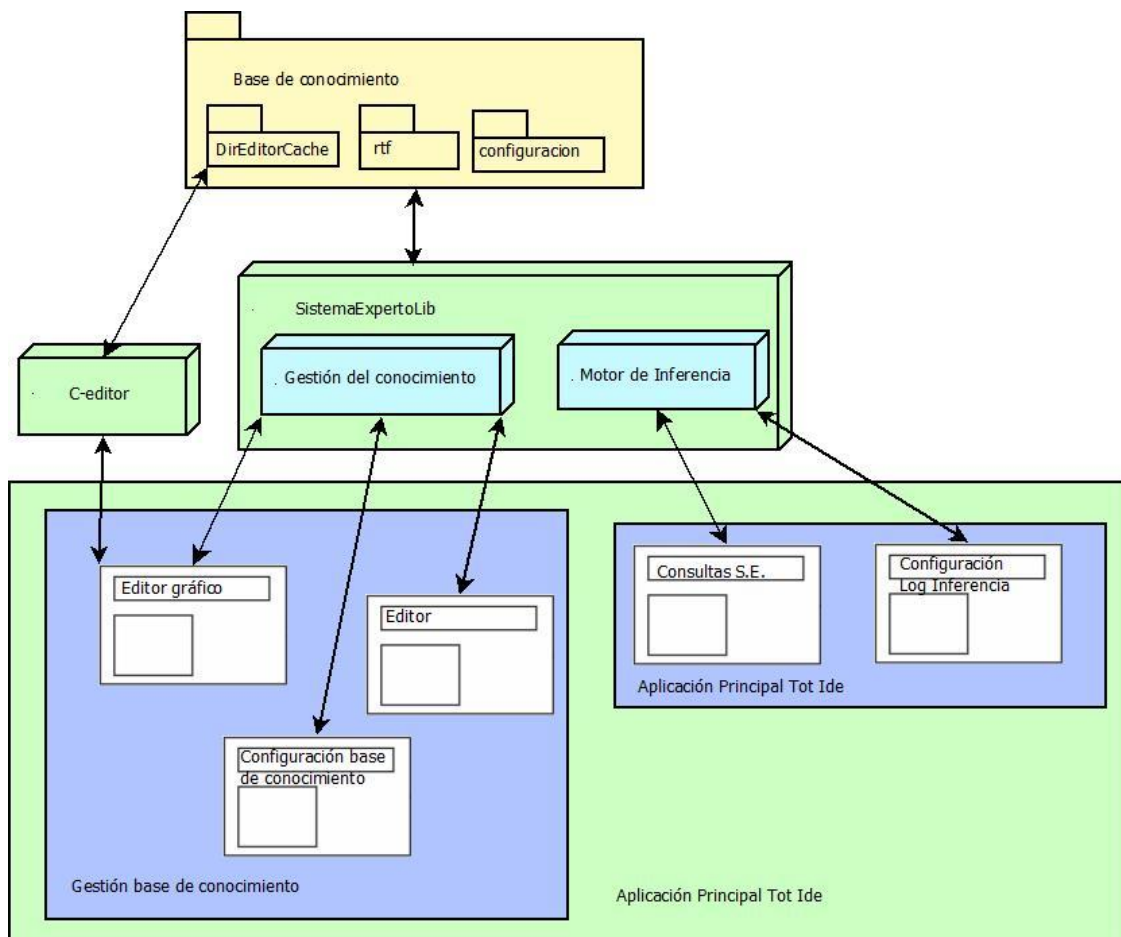
### **4.1. Arquitectura del Sistema**

Desde el punto de vista de estructural el software se compone de dos módulos principales el primero para la gestión de la base de conocimiento y el segundo para las consultas a las misma.

El módulo de gestión provee de las funcionalidades necesarias para la edición general de Variables y reglas de la base de conocimiento, además cuenta con un editor gráfico de reglas para exportar reglas a través de redes semánticas.

El módulo de consulta permite realizar los procesos de inferencia con la información almacenada en la base de conocimiento, para ir testeando a medida que se progresa en la misma.

### **4.2. Diagrama de Componentes**

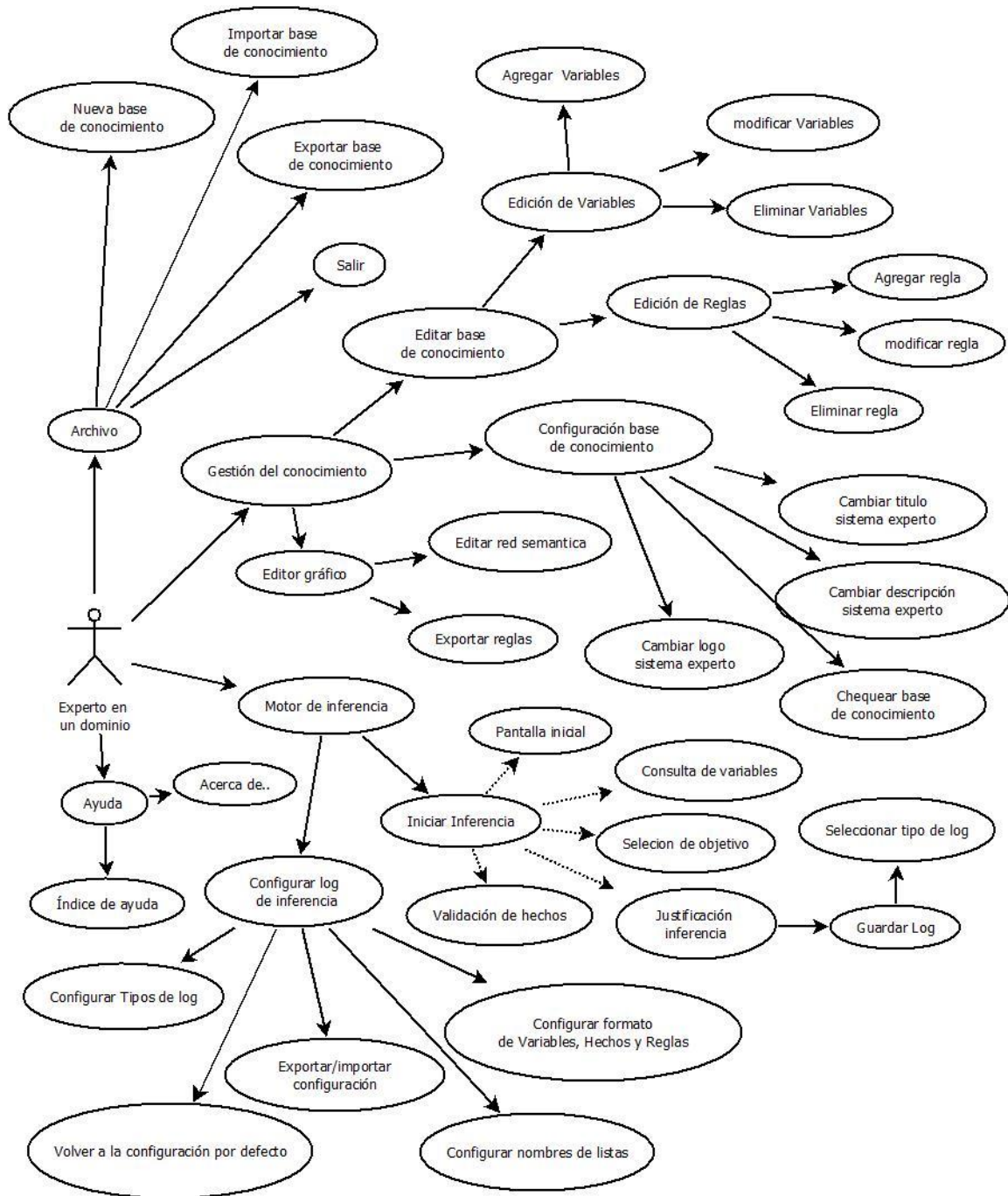


## 4.3. Diseño de Interfaz

### 4.3.1. Interfaz de Usuario

Del diagrama de casos de uso podemos definir las funcionalidades que deben estar disponibles desde la interfaz de usuario.



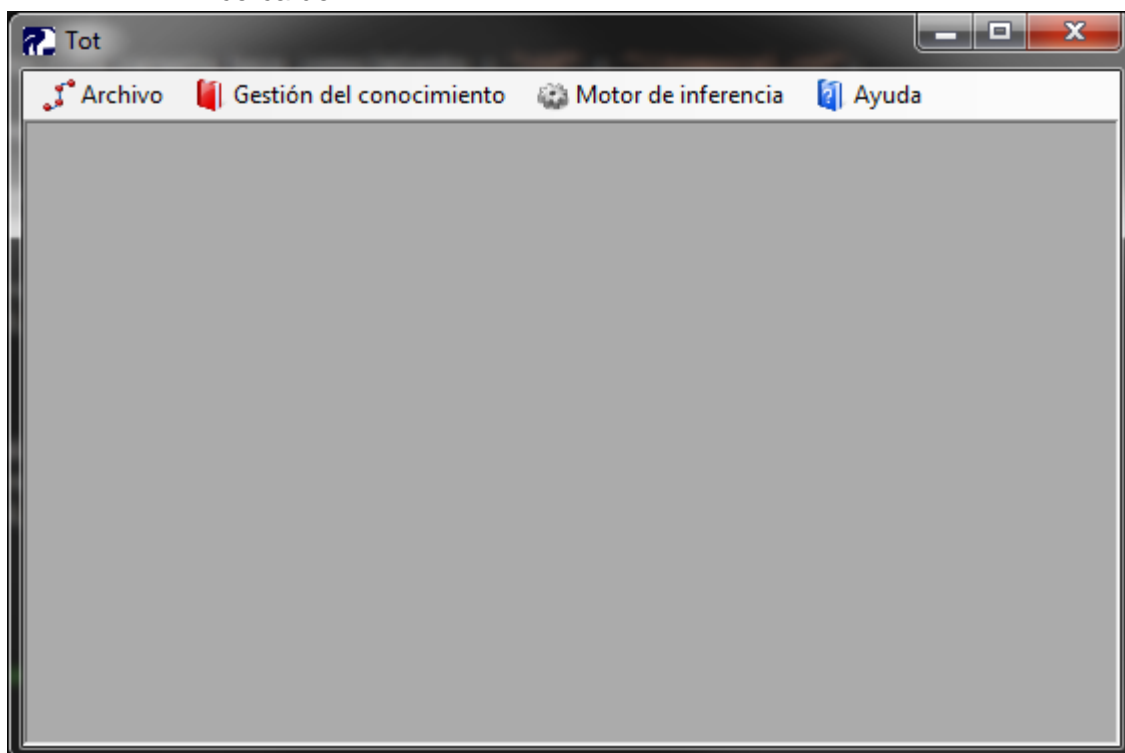


### 4.3.2. Ventana principal

La ventana principal tiene el link con las principales funcionalides de la aplicacion

- Archivo
  - Nueva base de conocimiento
  - Importar base de conocimiento
  - Exporta base de conocimiento

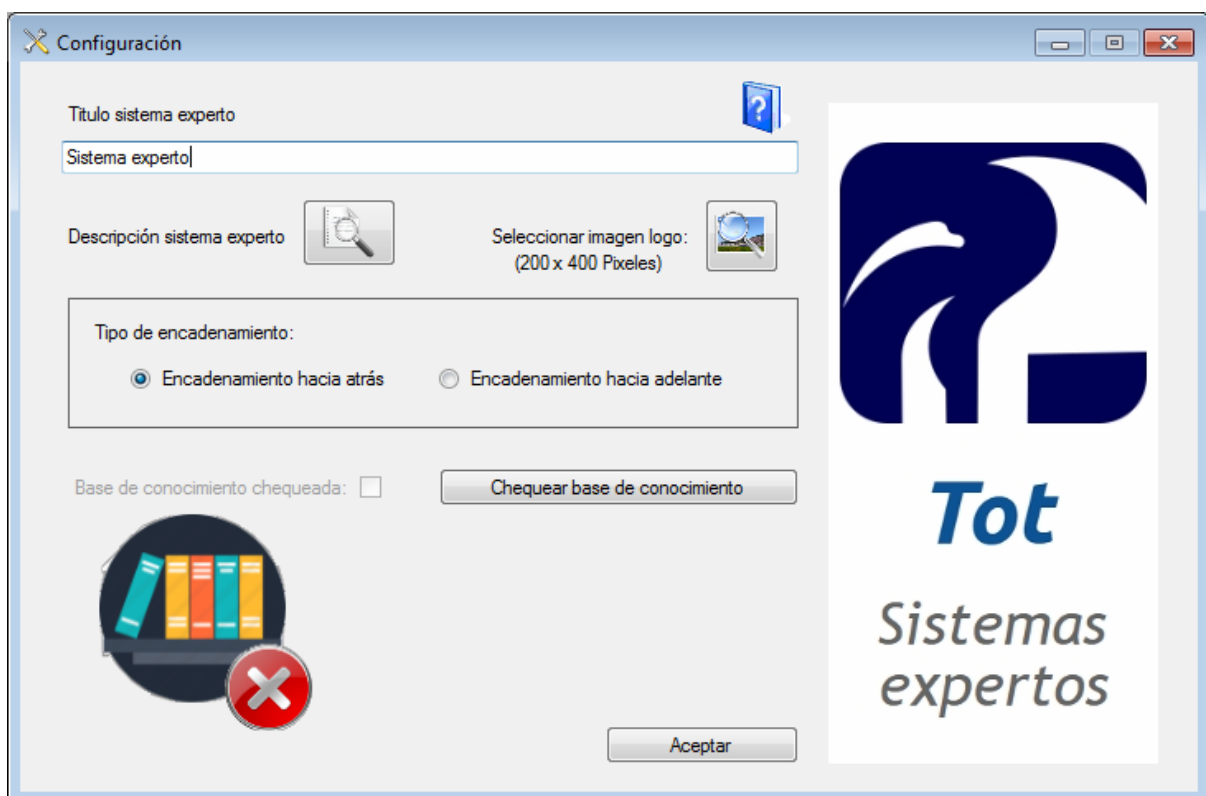
- Salir
- Gestion del conocimiento
  - Editar base de conocimiento
  - Editor gráfico
  - Configurar base de conocimiento
- Motor de inferencia
  - Iniciar inferencia
  - Configurar log de inferencia
- Ayuda
  - Indice de ayuda
  - Acerca de..



### 4.3.3. Ventana configuración

En esta ventana se pueden realizar las configuraciones de la base de conocimiento

- Funcionalidades
  - Cambiar título sistema experto
  - Cambiar descripción sistema experto
  - Cambiar logo sistema experto
  - Chequear base de conocimiento



### 4.3.4. Ventana Gestión de la base de conocimiento

En esta ventana se pueden realizar las configuraciones de la base de conocimiento, cuenta con tres paneles disponibles :

#### 4.3.4.1. Panel variables

En este panel se puede realizar la gestión de las variables en general, Ingreso, modificación y eliminación de las variables.

- Funcionalidades
  - Agregar variable
  - Modificar variable
  - Eliminar variable

Gestión Base de Conocimiento

Variables Hechos Reglas

Lista de variables

(L) V_45	abrazaderas manguera del agua	[?]
(B) V_48	aceite en agua motor	[?]
(L) V_23	aceite motor	[?]
(L) V_24	afinamiento	
(B) V_49	agua sale a presion del radiador	[?]
(L) V_21	alternador	
(L) V_30	anillos piston	[?]
(B) V_11	apertura bujias	[?]
(B) V_1	auto funcional	[0]
(L) V_7	bateria	
(L) V_29	biela	[?]
(L) V_46	bomba de agua	
(L) V_9	bujias	
(L) V_12	cableado	[?]
(L) V_18	caja de cambio	[0]
(B) V_10	chispa bujias	[?]
(L) V_26	cigüeñal	[?]
(L) V_47	circulacion de agua	[?]
(L) V_14	correa distribucion	[?]
(N) V_13	corriente generada por alternador	[?]
(L) V_36	disco de embrague	[?]
(L) V_50	empaquetadura de culata	
(L) V_42	filtracion de agua radiador	[?]
(L) V_34	fuerza vehiculo	[?]
(L) V_20	fusibles	[?]
(L) V_43	mangueras del agua	
(L) V_38	marchas vehiculo	[?]
(L) V_16	motor	[0]
(B) V_22	motor andando	[?]
(L) V_25	motor interno	
(L) V_37	prensa (embrague)	[?]
(L) V_41	radiador	
(L) V_31	retenes válvula	[?]
(L) V_35	rodamiento empuje	[?]
(L) V_44	roturas mangueras del agua	[?]
(L) V_32	sistema de embrague	
(L) V_19	sistema de enfriamiento	[0]
(L) V_33	sistema de engranaje caja de cambio	
(L) V_17	sistema electrico	[0]
(L) V_39	sonidos caja de cambio	[?]
(L) V_6	tablero vehiculo	[1][?]
(L) V_40	temperatura motor	[?]

Detalle Variable

Id

Nombre

☐ Variable preguntable al usuario

☐ Variable de inicio (Encadenamiento hacia adelante)

☐ Variable Objetivo (Encadenamiento hacia atrás)

Tipo de Variable

☐ Booleano

☐ Numérico

☐ Lista de elementos

Texto consulta

Descripción específica de la variable

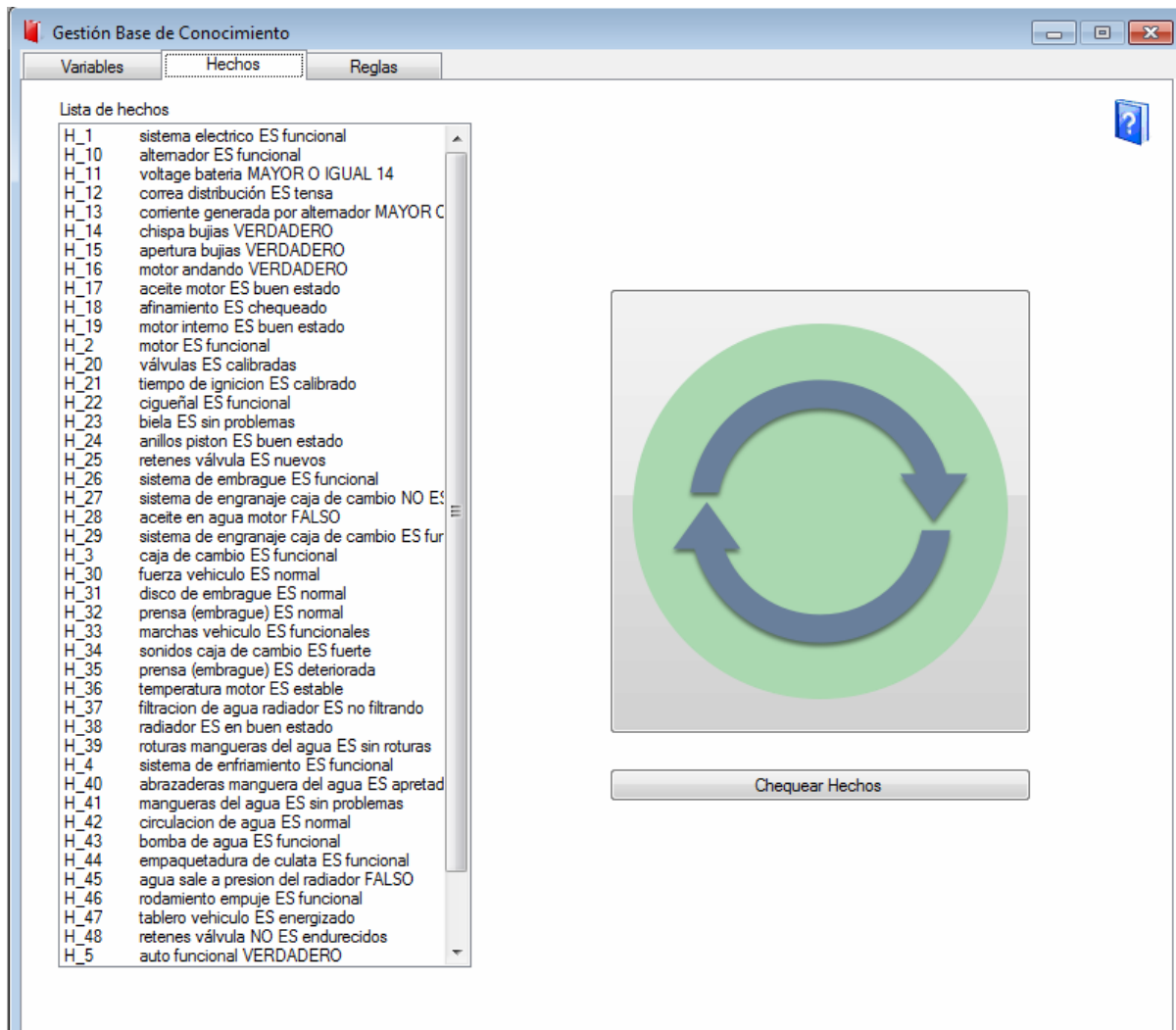
Eliminar Variable

Modificar Variable

Agregar Variable

#### 4.3.4.2. Panel Hechos

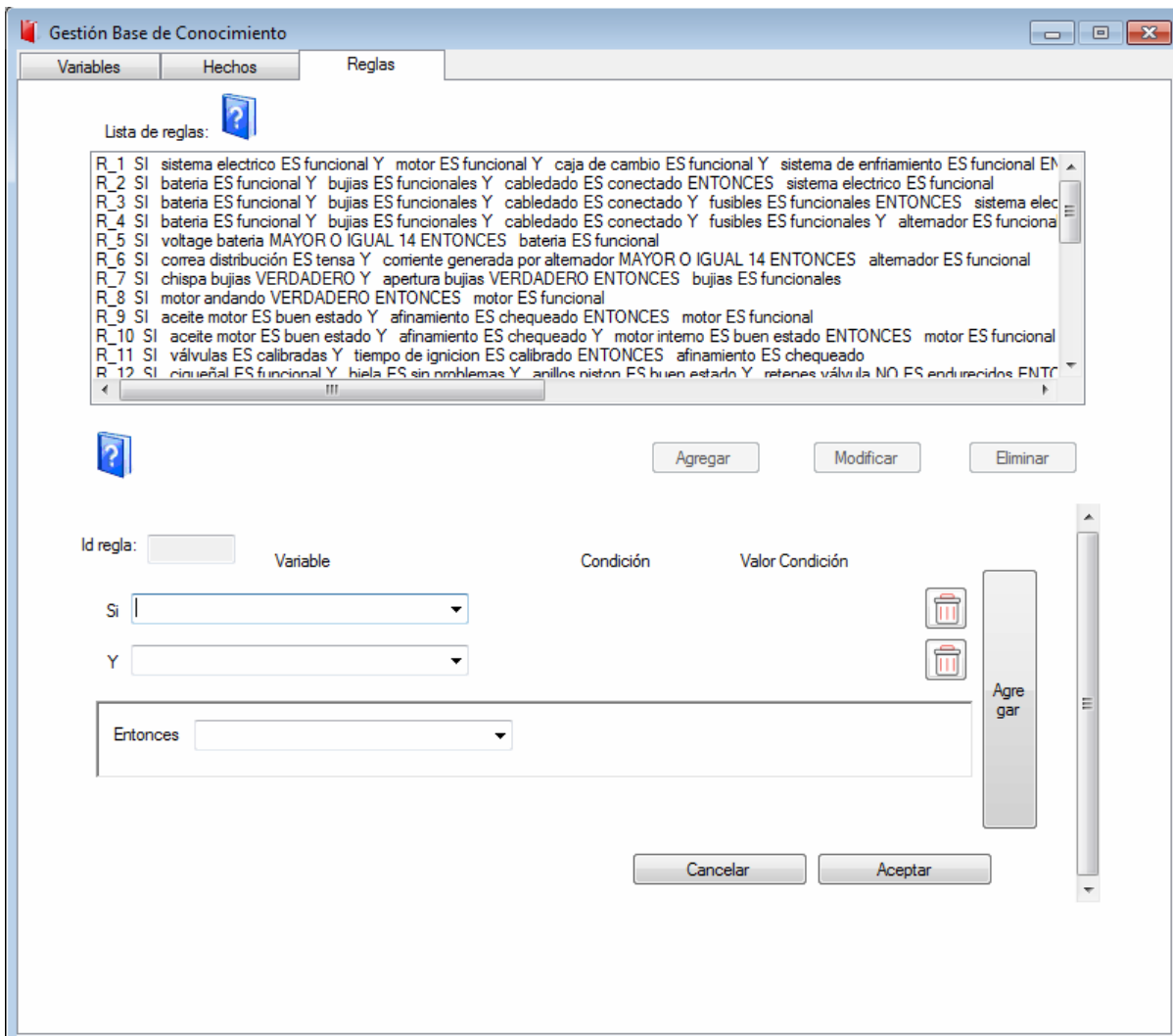
En este panel es sólo informativo y muestra la lista de hechos que se encuentran actualmente en la base de conocimiento



#### 4.3.4.3. Panel Reglas

En este panel se puede realizar la gestión general de las reglas de la base de conocimiento ingreso, modificación y eliminación de reglas.

- Funcionalidades
  - Creación de reglas
  - Modificar reglas
  - Eliminar reglas



**Gestión Base de Conocimiento**

Variables Hechos Reglas

Lista de reglas:

- R\_1 SI sistema electrico ES funcional Y motor ES funcional Y caja de cambio ES funcional Y sistema de enfriamiento ES funcional EN
- R\_2 SI bateria ES funcional Y bujias ES funcionales Y cableado ES conectado ENTONCES sistema electrico ES funcional
- R\_3 SI bateria ES funcional Y bujias ES funcionales Y cableado ES conectado Y fusibles ES funcionales ENTONCES sistema elec
- R\_4 SI bateria ES funcional Y bujias ES funcionales Y cableado ES conectado Y fusibles ES funcionales Y alternador ES funciona
- R\_5 SI voltage bateria MAYOR O IGUAL 14 ENTONCES bateria ES funcional
- R\_6 SI correa distribución ES tensa Y corriente generada por alternador MAYOR O IGUAL 14 ENTONCES alternador ES funcional
- R\_7 SI chispa bujias VERDADERO Y apertura bujias VERDADERO ENTONCES bujias ES funcionales
- R\_8 SI motor andando VERDADERO ENTONCES motor ES funcional
- R\_9 SI aceite motor ES buen estado Y afinamiento ES chequeado ENTONCES motor ES funcional
- R\_10 SI aceite motor ES buen estado Y afinamiento ES chequeado Y motor interno ES buen estado ENTONCES motor ES funcional
- R\_11 SI válvulas ES calibradas Y tiempo de ignición ES calibrado ENTONCES afinamiento ES chequeado
- R\_12 SI cincha ES funcional Y biela ES sin problemas Y anillos piston ES buen estado Y retenes válvula NO ES endurecidos ENT

Id regla:  Variable  Condición  Valor Condición

Si

Y

Entonces

Agregar Modificar Eliminar

Cancelar Aceptar



### 4.3.5. Ventana configuración log de inferencia

En esta ventana provee de los campos necesarios para configurar el log de inferencia

- Funcionalidades
  - Configuración formato de Variables, Hechos y Reglas
  - Configurar nombres de lista
  - Configurar tipos de log
  - Importar / exportar configuración del log
  - Restablecer a la configuración por defecto

**Configuración log de inferencia**

Opciones de configuración: ?

Eliga el formato a mostrar en el log de inferencia:

Variables:	Hechos:	Reglas:
<input type="radio"/> ID	<input type="radio"/> ID	<input type="radio"/> ID
<input type="radio"/> NOMBRE	<input checked="" type="radio"/> CONDICIÓN	<input type="radio"/> REGLA
<input checked="" type="radio"/> ID + NOMBRE	<input type="radio"/> ID + CONDICIÓN	<input checked="" type="radio"/> ID + REGLA

Configuración de nombre de listas:

Reglas disponibles:	Reglas Disponibles
Reglas candidatas:	Reglas Candidatas
Reglas eliminadas:	Reglas Eliminadas
Hechos disponibles:	Hechos Disponibles
Hechos verdaderos:	Hechos Verdaderos
Hechos falsos:	Hechos Falsos

Tipo log: LOG\_VARIABLE

Mostrar ☒

Importar configuración Exportar configuración Configuración por defecto

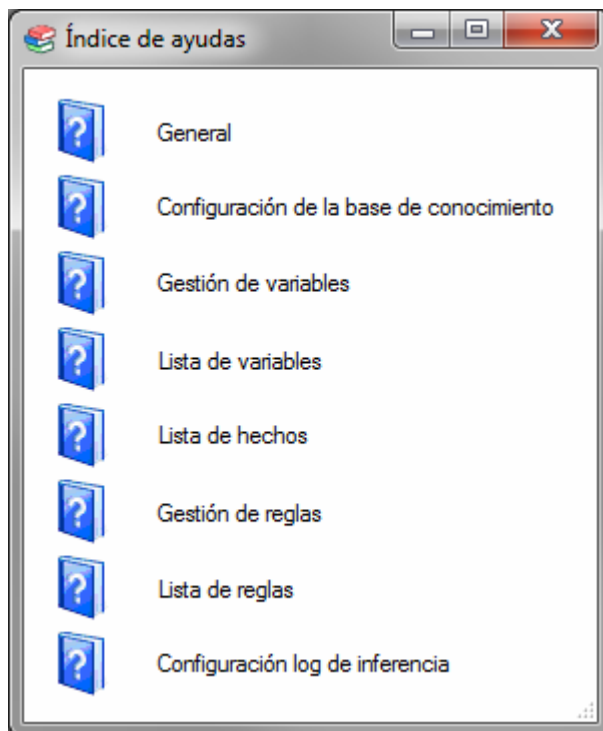
Cancelar Aceptar





#### 4.3.6. Ventana configuración log de inferencia

En esta ventana contiene los links a las ayudas del sistema.





### 4.3.7. Consulta Sistema experto

Mientras se realiza un proceso de inferencia se utilizan el siguiente set de diálogos

#### 4.3.7.1. Dialogo inicial

En esta ventana se da la bienvenida al usuario y se muestra la información descriptiva del sistema experto al usuario.





#### 4.3.7.2. Dialogo selección de objetivo

En el caso de encadenamiento hacia atrás, mediante este dialogo se selecciona el objetivo principal a comprobar en la base de conocimiento.

**Mecánica general...**

Eliga objetivo encadenamiento

- ☒ sistema electrico
- ☐ motor
- ☐ caja de cambio
- ☐ sistema de enfriamiento
- ☐ auto funcional

Seleccione el estado a comprobar

- ☒ sistema electrico ES funcional

**Iniciar Inferencia**



#### 4.3.7.3. Dialogo Consulta de usuario

Cuando el motor de inferencia requiere el valor de una variable por parte del usuario, esta petición se realiza a través de este dialogo. Aqui se muestra el nombre de la variable, la pregunta correspondiente, la descripción de la misma y por último el campo para ingresar el valor.

X

tablero vehiculo

¿Le llega energia electrica al tablero del vehiculo?



Ingrese estado variable: 

energizado

Continuar

#### 4.3.7.4. Dialogo validación de regla

Cuando el motor de inferencia requiere validar una regla se muestra el siguiente dialogo en donde se puede ver:

- El nombre de la variable asociada al hecho consecuente
- Los antecedentes de la regla
- El consecuente de la regla
- Las opciones para validar la regla
- Las opciones para marcar el problema como solucionado o no solucionado
- La opción de detener o seguir con el proceso de inferencia



bateria

La bateria es la fuente de energía eléctrica del automóvil.



Según R\_5

Como:

voltage bateria MAYOR O IGUAL 14

Se infiere que :

bateria ES funcional

¿Validar afirmación?

Si

No

¿Se solucionó el problema?

Si

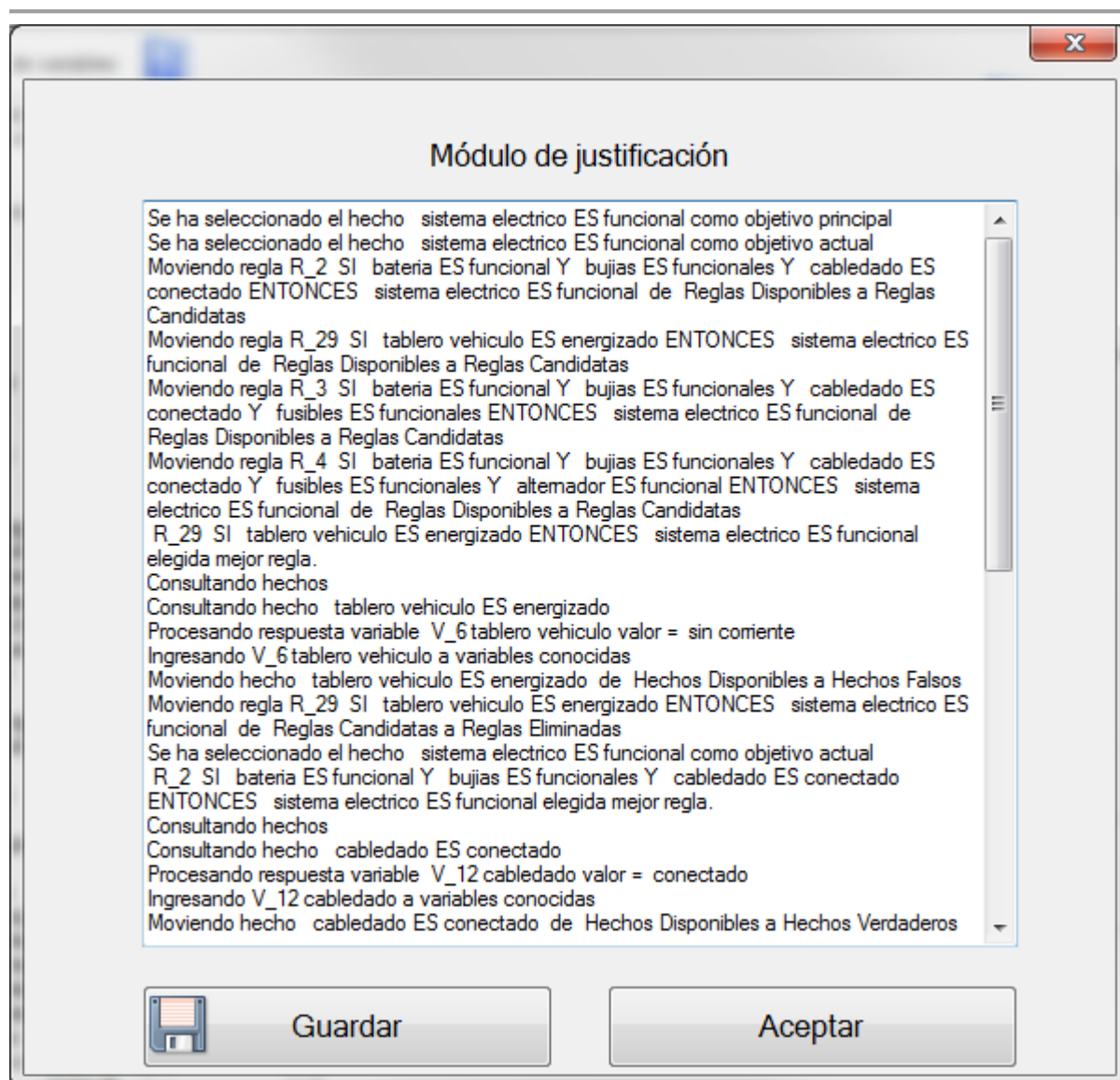
No

Detener Inferencia

Continuar Inferencia

#### 4.3.7.5. Dialogo validación de regla

Una vez terminado el proceso de inferencia se aparece el dialogo de justificación, en donde se muestra el log del proceso según lo haya configurado el usuario. Teniendo la opción de guardar el log en un archivo de texto

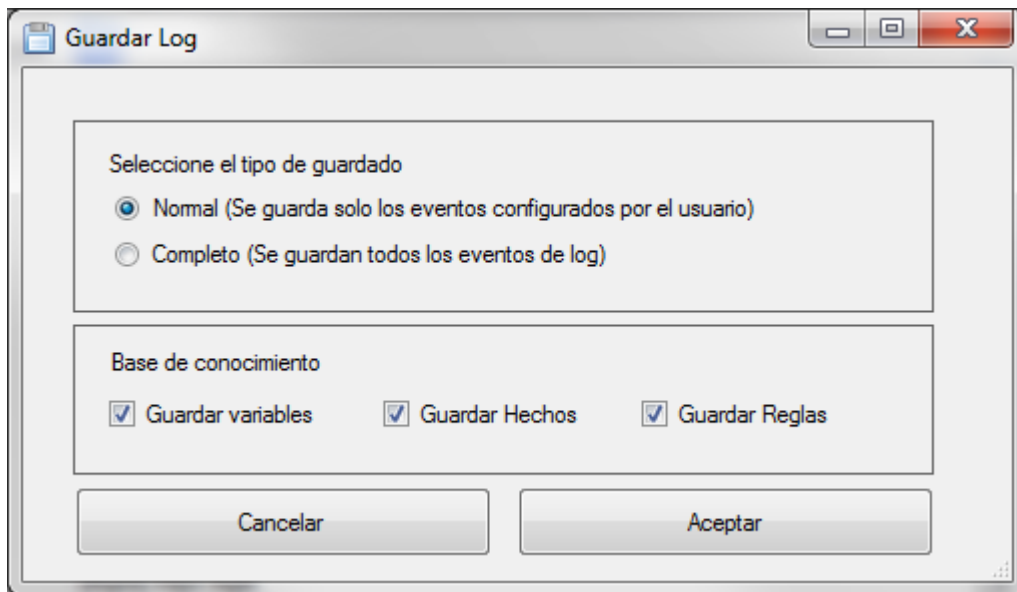


#### 4.3.7.6. Dialogo guardar log

Al momento de guardar se puede configurar la salida del archivo según los siguientes parámetros:



- Tipo de guardado
- Normal: Se guarda el log según lo configuró el usuario
- Completo: Se guardan todos los eventos de log, sin importar la configuración
- Base de conocimiento: Se puede anexar al log la información de variables, hechos y reglas según estime el usuario.





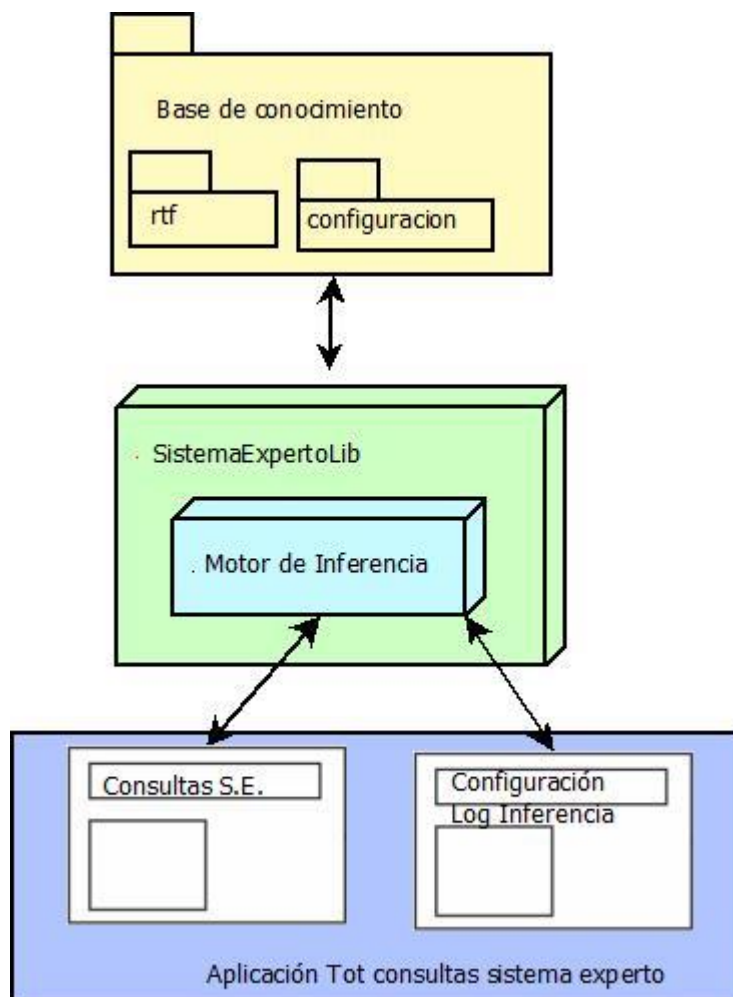


## 5. Módulo: Aplicación Tot Consultas sistema experto

### 5.1. Arquitectura del Sistema

Este software se compone solo del módulo de consulta de la base de conocimiento y brinda las funcionales para realizar los proceso de inferencia y guardar los resultados del log, además de permitir la configuraciones generales del log.

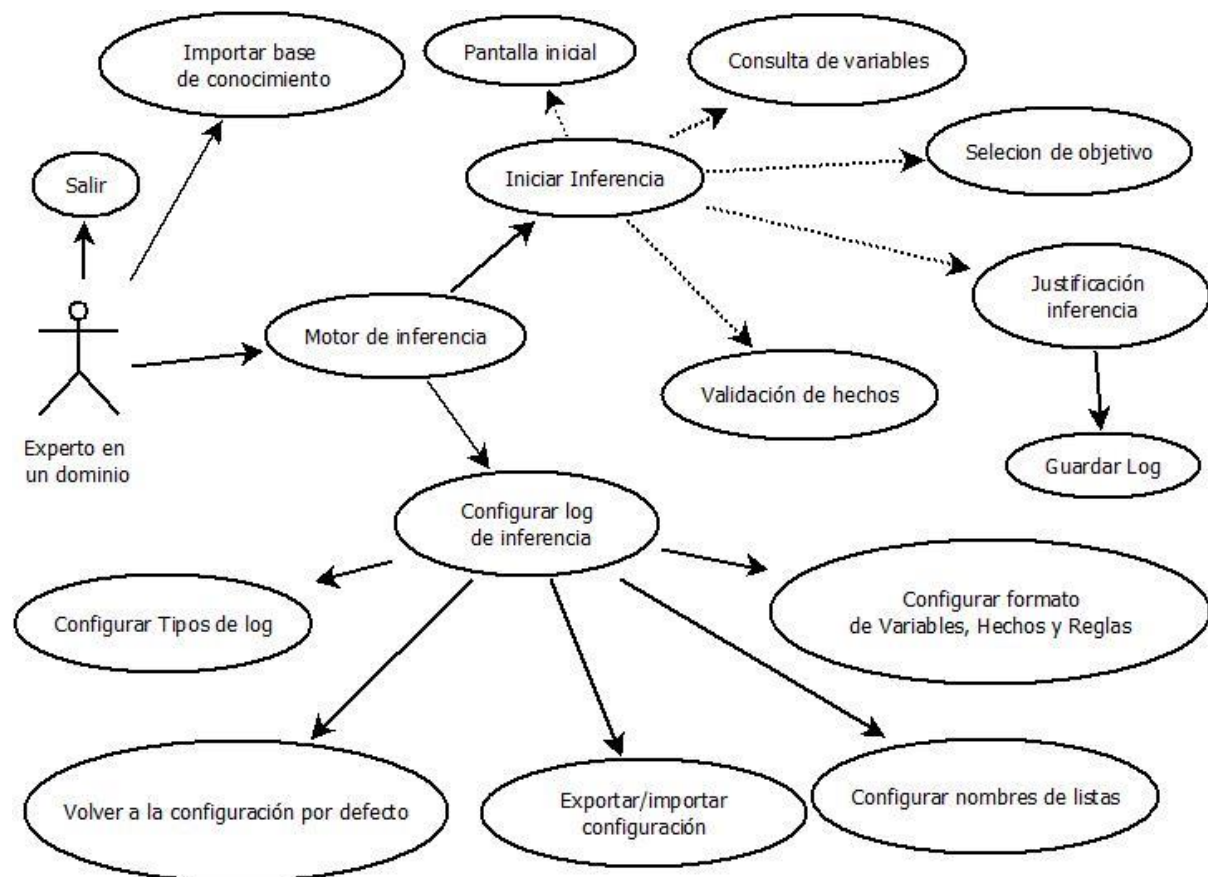
#### 5.1.1. Diagrama de Componentes



## 5.2. Diseño de Interfaz

### 5.2.1. Interfaz de Usuario

Del diagrama de casos de uso podemos definir las funcionalidades que deben estar disponibles desde la interfaz de usuario.





### 5.2.2. Ventana principal

La ventana principal tiene el link a las funciones de consulta del sistema experto:

- Importar base de conocimiento
- Configuración log de inferencia (Para mostrar el botón de configuración se debe clicar en reiteradas oportunidades la ventana)
- Iniciar proceso de inferencia



### 5.2.3. Ventana configuración log de inferencia

En esta ventana provee de los campos necesarios para configurar el log de inferencia

- Funcionalidades
  - Configuración formato de Variables, Hechos y Reglas
  - Configurar nombres de lista
  - Configurar tipos de log
  - Importar / exportar configuración del log (Se puede utilizar la misma configuración que la aplicación Tot IDE)
  - Restablecer a la configuración por defecto

**Configuración log de inferencia**

Opciones de configuración: ?

Eliga el formato a mostrar en el log de inferencia:

Variables:	Hechos:	Reglas:
<input type="radio"/> ID	<input type="radio"/> ID	<input type="radio"/> ID
<input type="radio"/> NOMBRE	<input checked="" type="radio"/> CONDICIÓN	<input type="radio"/> REGLA
<input checked="" type="radio"/> ID + NOMBRE	<input type="radio"/> ID + CONDICIÓN	<input checked="" type="radio"/> ID + REGLA

Configuración de nombre de listas:

Reglas disponibles:	Reglas Disponibles
Reglas candidatas:	Reglas Candidatas
Reglas eliminadas:	Reglas Eliminadas
Hechos disponibles:	Hechos Disponibles
Hechos verdaderos:	Hechos Verdaderos
Hechos falsos:	Hechos Falsos

Tipo log: LOG\_VARIABLE

Mostrar ☒

Importar configuración Exportar configuración Configuración por defecto

Cancelar Aceptar





### 5.2.4. Consulta Sistema experto

Mientras se realiza un proceso de inferencia se utiliza el siguiente set de diálogos

#### 5.2.4.1. Dialogo inicial

En esta ventana se da la bienvenida al usuario y se muestra la información descriptiva del sistema experto al usuario.





#### 5.2.4.2. Dialogo selección de objetivo

En el caso de encadenamiento hacia atrás, mediante este diálogo se selecciona el objetivo principal a comprobar en la base de conocimiento.

The dialog box is titled "Mecánica general..." and contains two main sections. The first section, "Eliga objetivo encadenamiento", has a list of five radio buttons: "sistema electrico" (selected), "motor", "caja de cambio", "sistema de enfriamiento", and "auto funcional". The second section, "Seleccione el estado a comprobar", has a single radio button labeled "sistema electrico ES funcional" which is also selected. At the bottom is a large button labeled "Iniciar Inferencia".

Mecánica general...

Eliga objetivo encadenamiento

- ☒ sistema electrico
- ☐ motor
- ☐ caja de cambio
- ☐ sistema de enfriamiento
- ☐ auto funcional

Seleccione el estado a comprobar

- ☒ sistema electrico ES funcional

Iniciar Inferencia



#### 5.2.4.3. Dialogo Consulta de usuario

Cuando el motor de inferencia requiere el valor de una variable por parte del usuario, esta petición se realiza a través de este diálogo. Aquí se muestra el nombre de la variable, la pregunta correspondiente, la descripción de la misma y por último el campo para ingresar el valor.

X

tablero vehiculo

¿Le llega energia eletrica al tablero del vehiculo?



Ingrese estado variable: 

energizado

Continuar



#### 5.2.4.4. Dialogo validación de regla

Cuando el motor de inferencia requiere validar una regla se muestra el siguiente dialogo en donde se puede ver:

- El nombre de la variable asociada al hecho consecuente
- Los antecedentes de la regla
- El consecuente de la regla
- Las opciones para validar la regla
- Las opciones para marcar el problema como solucionado o no solucionado
- La opción de detener o seguir con el proceso de inferencia

bateria

La bateria es la fuente de energía eléctrica del automóvil.



Según R\_5

Como:

voltage bateria MAYOR O IGUAL 14

Se infiere que :

bateria ES funcional

¿Validar afirmación?

Si

No

¿Se solucionó el problema?

Si

No

Detener Inferencia

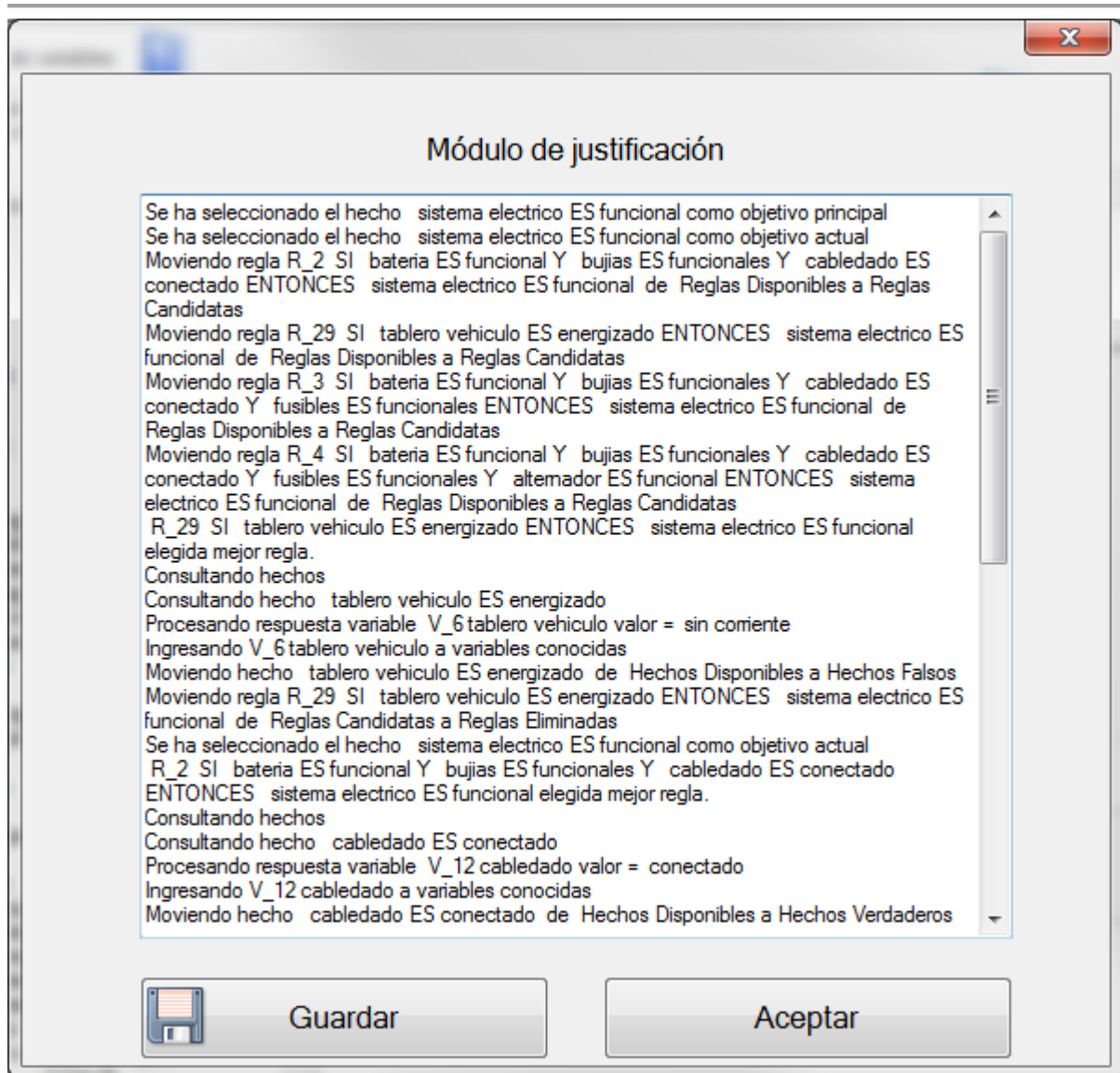
Continuar Inferencia

#### 5.2.4.5. Dialogo validación de regla

Una vez terminado el proceso de inferencia se aparece el diálogo de justificación, en donde se muestra el log del proceso según lo haya configurado el usuario. Teniendo la opción de guardar el log en un archivo de texto

Ingeniería de Software 1 - 2016

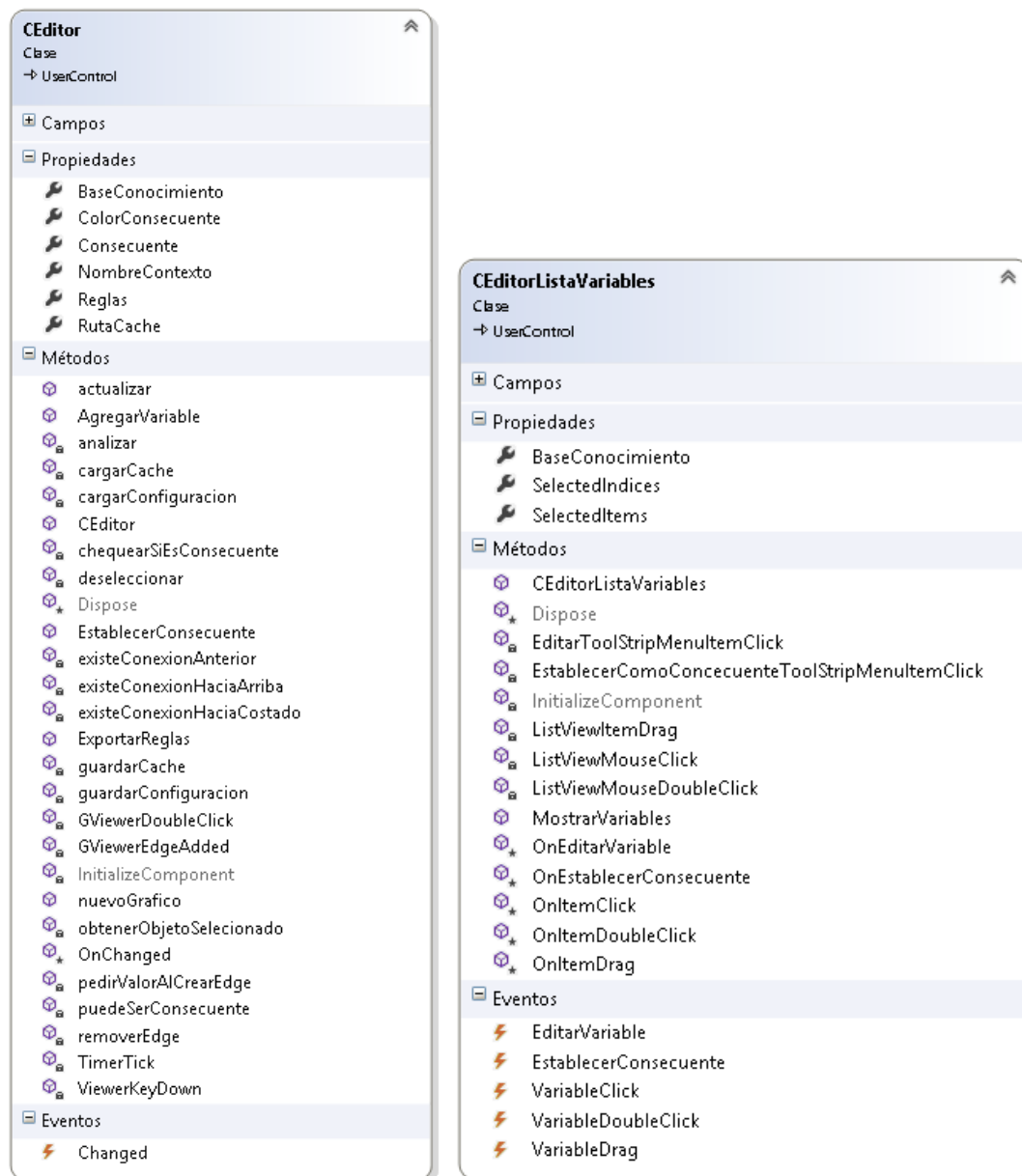
81



## 6. Módulo: Editor Gráfico de reglas - Control de usuario C-Editor.

### 6.1. Arquitectura

El objetivo de este módulo es proveer de un control de usuario para el ingreso en forma gráfica de reglas a la base de conocimiento.



El control de Usuario *CEditor* es un diseñador gráfico para las reglas. en él se puede dibujar cada antecedente de una regla como nodos de un grafo dirigido.

El control de usuario *CEditorListaVariable* se encarga de cargar las variables presentes en la base de conocimiento, con este control podemos agregar estas variables al gráfico.

