

Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

Videos Recomendados para su Visualización previa a la clase



Ejemplo SE Diagnóstico Médico	Simple SE Selección de Profesores (en CLIPS)	Ejemplo SE Errores en Computadores	Ejemplo Explicación Experto Diagnóstico

Tema 1: Ingeniería del Conocimiento

1.1: Ingeniería del Conocimiento. Representación del Conocimiento.

Ingeniería del Conocimiento: Conceptos y técnicas.

Representación del Conocimiento: Ingeniería Ontológica. Categorías de Objetos (marcos/frames).

Herramientas de desarrollo.

1.2: Razonamiento. Encadenamiento y Control.

Reglas.

Control Inferencial.

Encadenamiento inferencial. Forward & Backward.



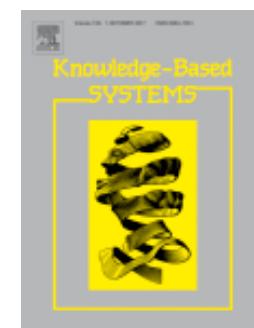
1.3: Sistemas Basados en el Conocimiento, Sistemas Expertos. Aplicaciones.

Definición, funcionalidades.

Entornos de desarrollo y áreas de aplicación.

Metodologías de desarrollo.

Aplicaciones (Sistemas Expertos). Extensiones.



Bibliografía

- **Inteligencia Artificial: Un enfoque moderno.** Russel, Norvig. Prentice Hall, 2004. (**Cap. 7, 10**)
- **Inteligencia Artificial. Técnicas, métodos y aplicaciones.** Varios autores. McGraw Hill (2008) **Cap.2-5, 19**
- **Inteligencia Artificial. Una nueva síntesis.** N. Nilsson. McGraw Hill (2000). (**Cap. 17**)
- **Poliformat: “Trends in expert system development: A longitudinal content analysis of over thirty years of expert system case studies”** W. P. Wagner . *Expert Systems With Applications* 76 (2017) 85–96

Ingeniería del Conocimiento. Conceptos y técnicas.

La **Ingeniería del Conocimiento** tiene como objetivo la **representación del conocimiento** (Base de Conocimiento) de forma **apta para ser procesable** mediante *procesos de razonamiento* (*Motor de Inferencias*)

Aplicaciones:

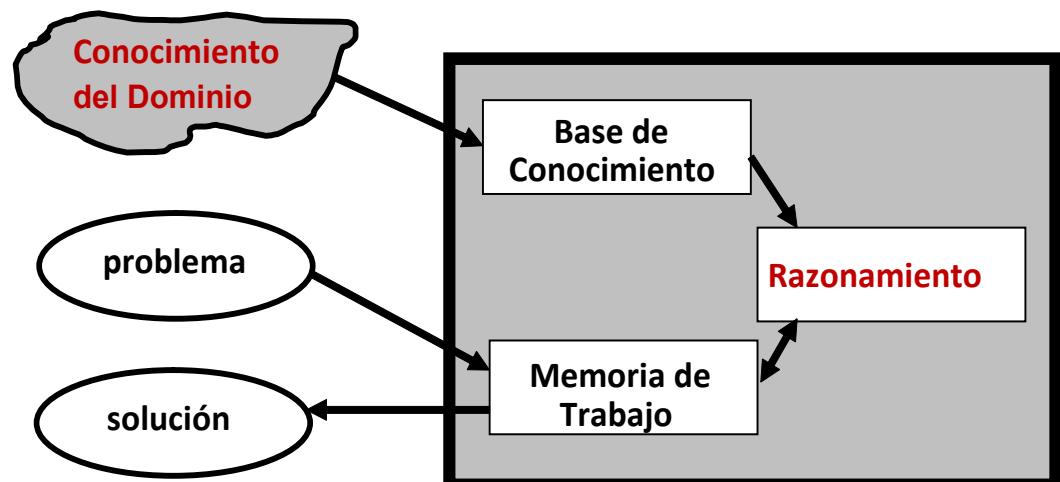
Sistemas Basados en el Conocimiento, Sistemas de Recomendación, Sistemas Basados en Casos (CBR), Mapas conceptuales, Web semántica, Ciencia de Datos, Business Intelligence, etc.

Sistema Basado en el Conocimiento

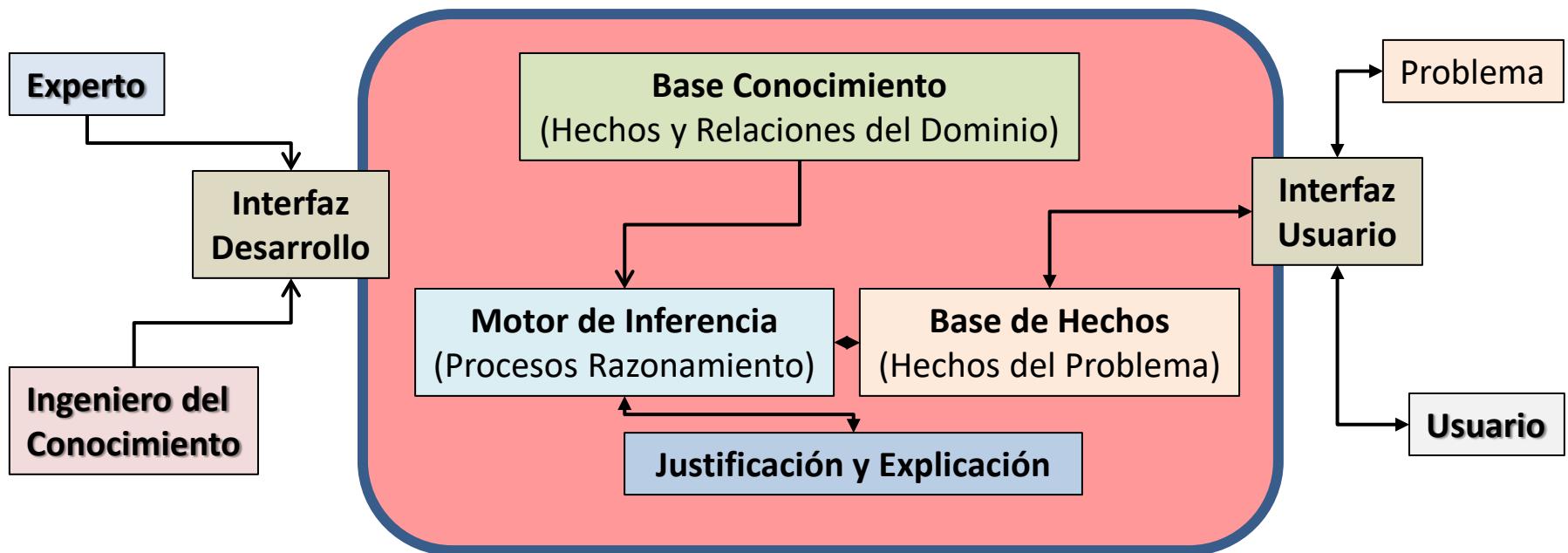
Sistema informático que

- tiene una **representación explícita del conocimiento** en algún dominio de competencia específico, y
- es capaz de explotarlo mediante **mecanismos de razonamiento** (técnicas de resolución de problemas)

con el fin de proporcionar **respuestas a problemas de alto nivel**.



Estructura de un Sistema Basado en el Conocimiento



Organización del conocimiento separada:

- Conocimiento del Dominio de aplicación (**BC**): Reutilizable, Permanente
- Datos del Problema (**BH**): Retractable, Dinámica.
- Resolución de problemas (**MI**): Control y Encadenamiento Inferencial.
- Interacción con el usuario, aprendizaje, etc.

Aplicación típica de la Ingeniería del Conocimiento (IA Simbólica):

- Representación y uso del Conocimiento para la resolución de problemas

Ingeniería del Conocimiento	Ingeniería del Software
Elección Método de Representación	Elección lenguaje programación
Conocimiento (Qué es verdad)	Datos (Qué es verdad) Algoritmo: Método (Cómo)
<p>Construcción de una BC</p> <ul style="list-style-type: none"> Conocimiento de control y de resolución de problemas por separado. Conocimiento representado principalmente de forma declarativa. 	<p>Elaboración de un programa</p> <ul style="list-style-type: none"> Conocimiento de control y de resolución de problemas integrado en el programa Conocimiento principalmente procedural. Sólo puede ser representado conocimiento algorítmico y estructurado
Inferencia de nuevos hechos	Ejecución del programa
<p>Modo de Operación no determinista</p> <ul style="list-style-type: none"> Razonamiento heurístico, cualitativo, incierto, impreciso, no algorítmico, etc. 	Modo de Operación determinista (metódico)
Transparente, Flexible	Opaco, Rígido, Algorítmico

1.1. Representación del Conocimiento en IA. Ingeniería Ontológica.



Representación de Hechos:

- a) Representación basada en la lógica (lógica proposicional/primer orden):

$$(\forall x), (\forall y) [<(x, 0) \vee <(y, 0) \vee >((x * y), 0)]$$

Formal, pero poco modular/ legible. No adecuados para dominios complejos.

- b) Representación mediante Hechos ordenados (similar visto en SIN - 3º)

(Hombre Nombre Adán Enfermedad Gripe Posee Barco1)

(Enfermedades Gripe Cólico Anemia)

(Mujer Nombre Eva Edad 21 Estudia Medicina)

(Enfermedad Gripe Síntomas Fiebre Dolor)

- No agrupación de conceptos/objetos, identificación de propiedades comunes, relaciones entre objetos, jerarquías o categorías (tipos/clases/subclases), etc.
- No adecuados para dominios complejos: Se requiere una estructura, generalizaciones, excepciones, flexibilidad, modularidad, etc.

- En dominios de “juguete” el problema de la representación es sencillo.
- En dominios reales y complejos requieren representaciones más estructuradas y flexibles.

Idea:

Identificar y Organizar los objetos (y elementos) del dominio a describir en Categorías

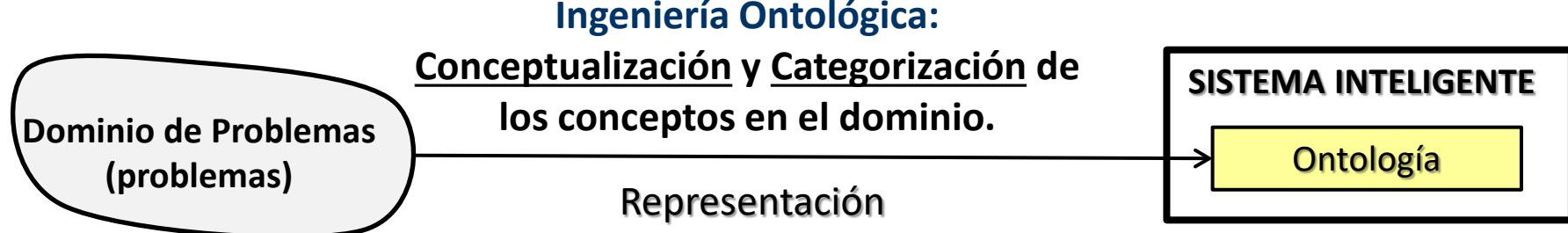
- La organización de los objetos (conceptos, etc.) a describir en categorías es muy útil:
el conocimiento suele describirse a nivel de categorías, más que a nivel de objetos concretos.

Es más usual describir las características de una enfermedad, de un conjunto o estrategia de acciones, de un conjunto de objetos físicos, etc. que la descripción de casos concretos.

- Las categorías sirven para organizar y estructurar el conocimiento.
Las relaciones de subclasificación organizan las categorías en taxonomías.
- Además permite utilizar un mecanismo de herencia.
- Las relaciones de entre las categorías pueden ser de diferentes tipos: temporales, pertenencias, estructurales, parte-de, tamaños, etc.

Representación del Conocimiento ⇒ Ingeniería Ontológica

Representación del Conocimiento: Ingeniería Ontológica



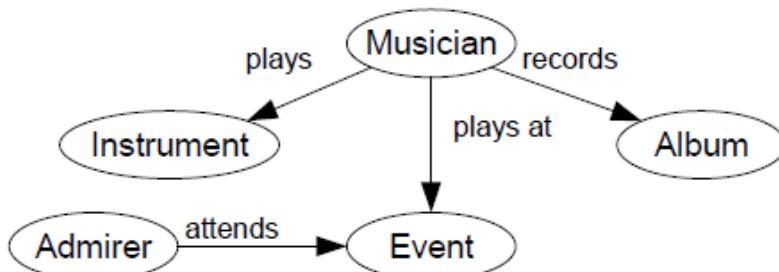
Ontología: Sistema particular de categorías que sistematiza una cierta visión del mundo

Componentes Básicos de una Ontología:

- **Clases y subclases:** Conceptos del dominio
 - Personas (hombre, mujeres).*
 - Tamaños (grande, pequeño, mediano)*
 - Vehículos (marítimos, terrestres, aéreos),*
 - Colores (azul, rojo, etc.), etc.*
- **Instancias:** Elementos concretos de la ontología.
 - Adán, Eva, etc.*
 - Talla-38*
 - Vehículo con matrícula 1234-TT*
 - Vuelo IB-537*
- **Relaciones n-arias** entre conceptos. Generalmente binarias: Relación (Concepto, Valor).
 - Posee (Persona, Objeto)*
 - Origen (Vuelo-537, Valencia)*
- **Axiomas:** Afirmaciones siempre ciertas en la ontología.
 - "un viaje no puede tener dos orígenes", "no se puede viajar a Mallorca en tren"*

Ejemplos de formalismos de representación de ontologías

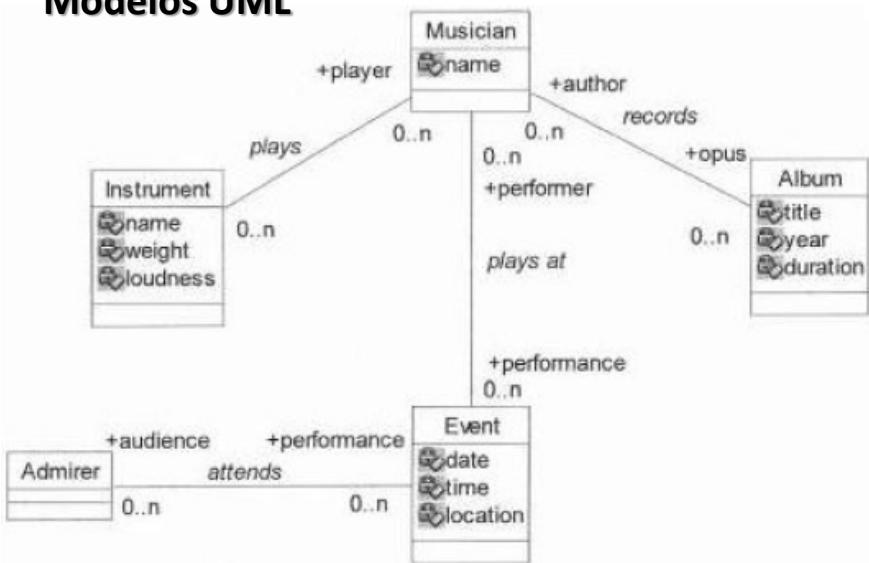
Redes Semánticas



Lenguaje OWL

```
<owl:Class rdf:ID="Event"/>
<owl:Class rdf:ID="Album"/>
<owl:Class rdf:ID="Instrument"/>
<owl:Class rdf:ID="Musician"/>
<owl:Class rdf:ID="Admirer"/>
<owl:ObjectProperty rdf:ID="author">
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="opus"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Album"/>
  <rdfs:range rdf:resource="#Musician"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="player">
  <rdfs:range rdf:resource="#Musician"/>
  <rdfs:domain rdf:resource="#Instrument"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="loudness">
  <rdf:type
    rdf:resource="http://www.w3.org/2007/07/owl#FunctionalProperty"/>
  <rdfs:domain rdf:resource="#Instrument"/>
</owl:ObjectProperty>
...
...
```

Modelos UML



UML (en ISW)	Objetos en IA (TIA)
Clases	Frames (marcos), clases o templates
Atributos con su tipo	Slots con sus facetas
Instancias	Instancias
Relaciones	Relaciones vía slots
Restricciones (OCL, CRDD, CRTT, etc.)	Axiomas para representar restricciones o afirmaciones

Frames/Templates: Representan (sub)Clases e Instancias (con Atributos o Slots, Facetas).

- Una **clase** representa conceptos, elementos genéricos o entidades con unas características comunes. Permite representar las características comunes (**propiedades, atributos o slots**) que tienen los objetos de una categoría, conjunto o abstracción. Algunas propiedades tendrán valor, otras no.

Clase: Personas

- Nombre
 Tipo: string, Cardinalidad: simple
- Edad
 Tipo: entero, Cardinalidad: simple
- Sexo
 Tipo: symbol, Cardinalidad: simple, Values {V, M}
- Teléfono
 Tipo: entero, Cardinalidad: múltiple

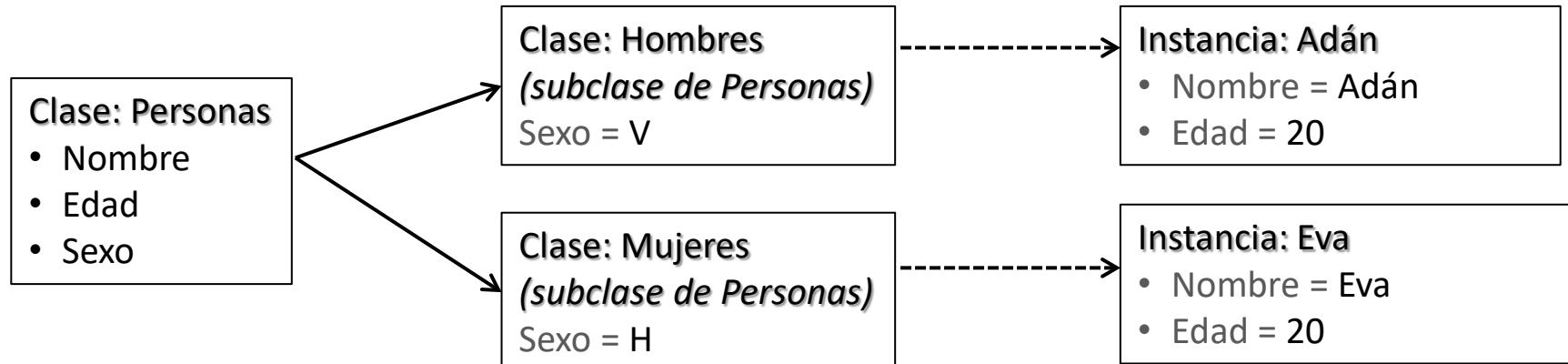
Los slots se caracterizan por sus **facetas o cualificadores**, típicamente:

- *Clase de Valores,*
 - *Cardinalidad,*
 - *Valor por defecto,*
 - *Valores permitidos...*
-
- Una **instancia** representa un objeto concreto.
 - Típicamente tiene **valores** en los slots, representando el conocimiento del objeto individual.

Instancia: Adán (*instancia de Hombres*)

- Nombre = Adán
- Edad = 20
- Sexo = V
- Teléfono = 123456 789012

En resumen. Componentes básicos de una ontología



- ✓ **Clases:** representan conceptos del dominio, en su sentido más amplio. Ej. persona, vehículo, etc.
- ✓ **Instancias:** Elementos concretos (individuos) de una clase. Ej. Adán, FordFiesta, etc.
- ✓ **Relaciones entre los conceptos:** Relaciones binarias: Relación (Concepto, Valor) modelados en los slots

Adán.Edad = 20 Hombres.Sexo = V

- ✓ **Axiomas:** Afirmaciones que son siempre ciertas en la ontología.

La herencia, jerarquía, tipo de valores, cardinalidad de slots, etc. imponen restricciones a la representación.

Ej: "una persona no puede tener dos edades" ⇒ Cardinalidad=1

Herramientas para Edición de Ontologías.

CLIPS-COOL (<https://www.clipsrules.net/>)

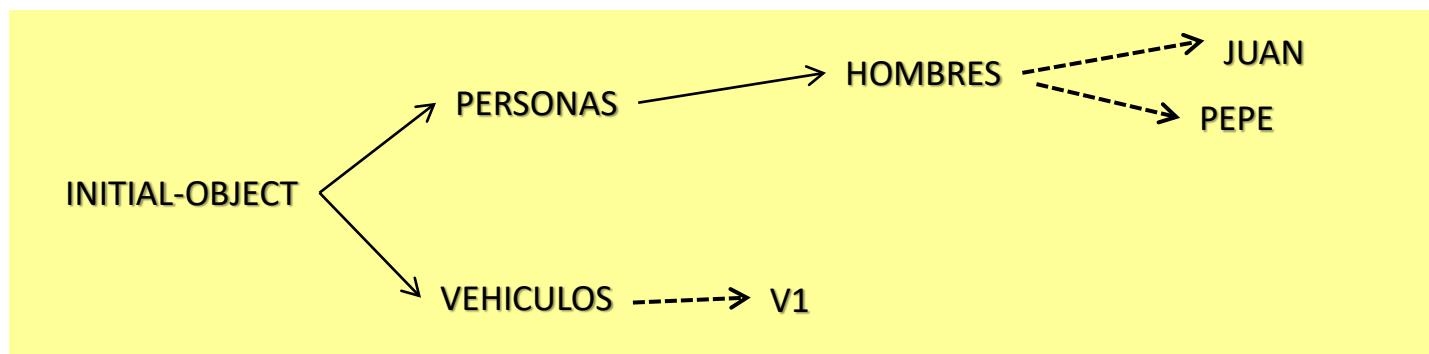
La extensión COOL de Clips (Clips Object Oriented Language) permite representar una ontología basada en categorías de objetos, mediante una **jerarquía de Clases e Instancias**.

```
(defclass PERSONAS (is-a INITIAL-OBJECT)
  (slot nombre (type STRING))
  (slot edad (type INTEGER) (range 20 40))
  (multislot apellidos (type STRING) (cardinality 2 2))
  (slot sexo (type SYMBOL)(allowed-values M H))
  (slot vehiculo (type INSTANCE-NAME)))
```

```
(defclass HOMBRES (is-a PERSONAS)
  (slot edad (source composite) (default 35))
  (slot sexo (source composite) (default H)))
```

```
(defclass VEHICULOS (is-a INITIAL-OBJECT)
  (slot matricula (type SYMBOL))
  (slot propietario (type INSTANCE-NAME)))
```

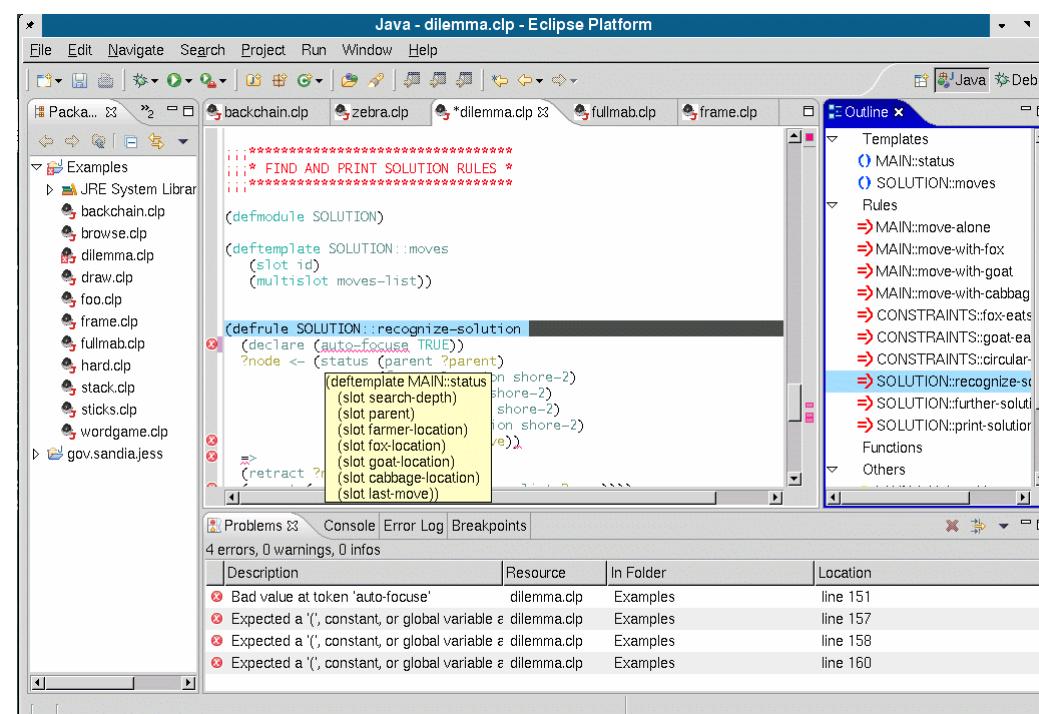
```
(definstances objetos
  (JUAN of HOMBRES (nombre "Oscar")(apellidos "Fernandez" "Lopez"))
  (V1 of VEHICULOS (matricula 3333CBS)(propietario [JUAN]))
  (PEPE of HOMBRES))
```



Herramientas para Edición de Ontologías.

Jess (<http://alvarestech.com/temp/fuzzyjess/Jess60/Jess70b7/docs/index.html>)

- Entorno de desarrollo de SBC como evolución de CLIPS. Libre para propósitos académicos
- JESS: escrito en Java. Utiliza la plataforma ECLIPSE como interfaz de edición
- Jess se puede integrar con Protégé mediante la extensión jesstab



(deftemplate persona

(slot nombre)

(slot direccion (type string))

(slot edad (type integer) (default 40))

(multislot hermanos (type symbol)))

(deftemplate estudiante extends persona

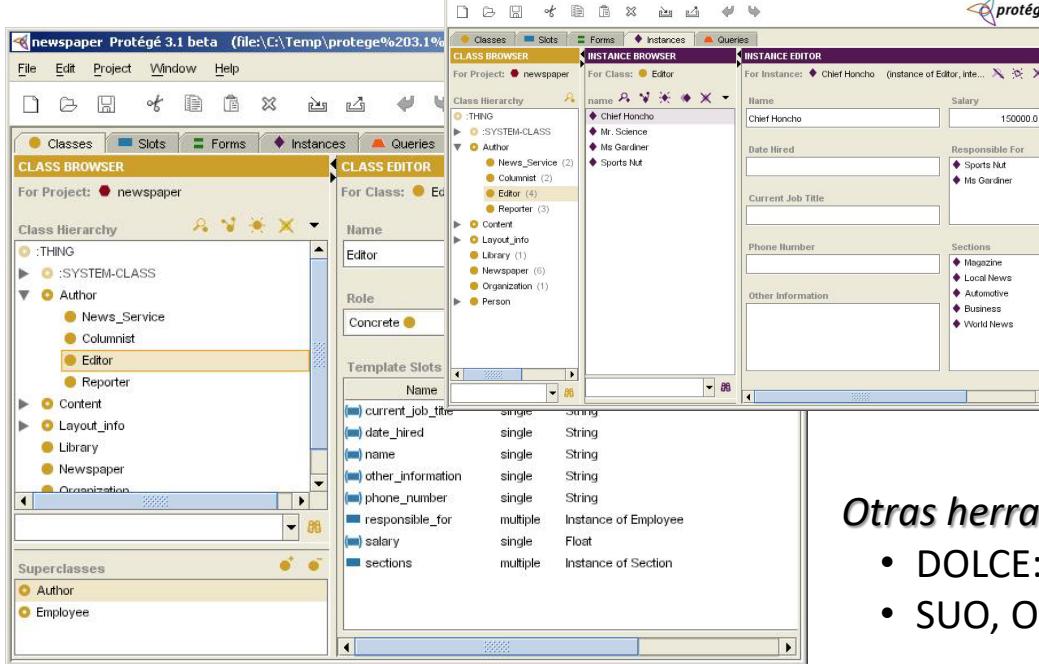
(slot edad (type integer) (default 20))

(slot titulacion))

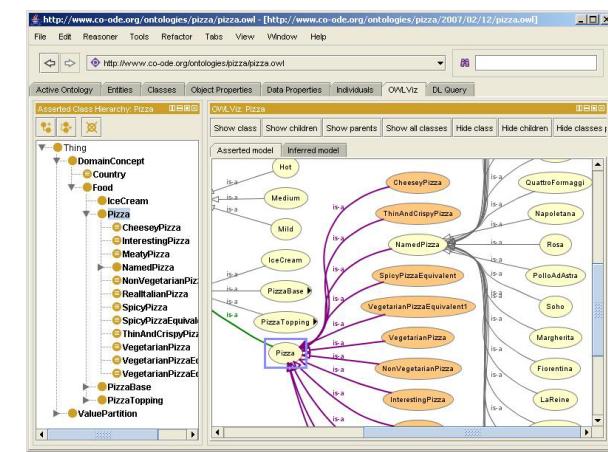
Herramientas gráficas para Edición de Ontologías basadas en Categorías

Protégé (<http://protege.stanford.edu/>)

- Herramienta para la construcción de modelos basados en ontologías:
 - Categorías de objetos: **Protégé-Frames**
 - Ontologías OWL (Web Ontology Language) para aplicaciones web semántica: **Protégé-OWL**
- Herramienta libre de código abierto
- Escrito en Java. Con demos web
- Proporciona una interfaz de usuario para edición
- No proporciona herramientas de ejecución o razonamiento:
enlace a CLIPS o Jess



Editor de clases



Interfaz Gráfico (OWL)

Editor de instancias

- Otras herramientas específicas para edición de ontologías:*
- DOLCE: <http://www.loa.istc.cnr.it/dolce/overview.html>
 - SUO, OntoEdit, Swoop, WebODE, etc.

Ejemplo Representar diversa información sobre la Universidad Politécnica de Valencia.

La Universidad está estructurada principalmente por Centros (responsables de las titulaciones) y Departamentos (a los que pertenecen los profesores de una o más áreas de conocimiento). Los profesores pueden impartir docencia en varias titulaciones y/o centros, pertenecen a un Departamento y están adscritos a un Centro. Un centro puede también impartir varias titulaciones. Particularmente, se debe representar:

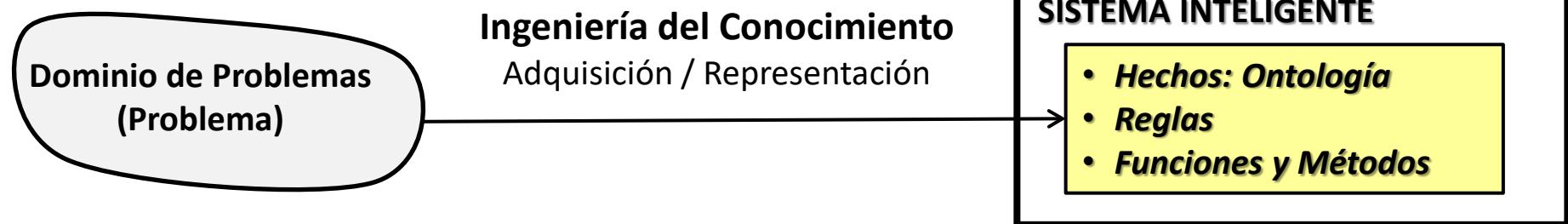
- Titulaciones que se cursan en la universidad.
- Centros Docentes, Titulaciones y Profesores adscritos a cada uno de ellos.
- Departamentos Universitarios, Profesores que lo forman y Titulaciones en las que el Departamento imparte Docencia.
- Los Alumnos y las Titulaciones en los que está matriculado.

Poner instancias de cada caso.

Ejemplo Representar diversa información sobre la Universidad Politécnica de Valencia.

La Universidad está estructurada principalmente por Centros (responsables de las titulaciones) y Departamentos (a los que pertenecen los profesores de una o más áreas de conocimiento). Los profesores pueden impartir docencia en varias titulaciones y/o centros, pertenecen a un Departamento y están adscritos a un Centro. Un centro puede también impartir varias titulaciones.

Representación del Conocimiento: Reglas (Conocimiento Normativo)



Las reglas representan conocimiento *heurístico* o *experimental*, que deben ser traducidas a reglas de producción (reglas físicas, causales, de experiencia, abductivas, etc.).

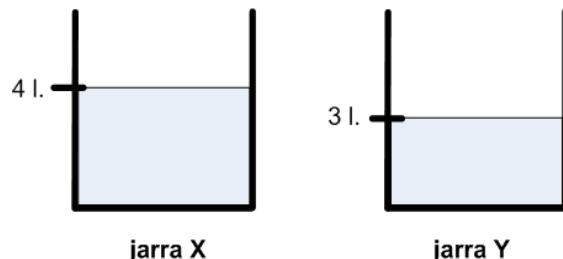
LHS \Rightarrow RHS

- Antecedente (**LHS**) representa **condiciones** sobre la BH, y se interpreta (**Pattern-matching**) sobre los hechos (BH, típicamente, condiciones sobre valores de atributos)
- Consecuente (**RHS**) realiza aserciones o modificaciones de valores de los slots de las instancias o acciones externas.

If
el paciente sufre dolor abdominal, y
se percibe murmullo abdominal en la auscultación, y
se palpa una masa pulsante en el abdomen
then
el paciente padece un aneurisma aórtico

La sintaxis de las reglas es dependiente del lenguaje de representación de los hechos.

Ejemplo Jarras (buscamos 2 litros):



(deftemplate JARRA
 (slot nombre (type SYMBOL))
 (slot litros (type INTEGER) (default 0))
 (slot capacidad (type INTEGER)))

Jerarquía de clases

(deftemplate JARRA-PEQUEÑA extends JARRA
 (slot capacidad (default 3)))

(deftemplate JARRA-GRANDE extends JARRA
 (slot capacidad (default 4)))

(deffacts jarras
 (JARRA-GRANDE (nombre J1))
 (JARRA-PEQUEÑA (nombre J2)))

Instancias existentes

(defrule llenar ;llenarjarra
?j <- (JARRA (nombre ?n) (capacidad ?cap) (litros ?lit))
 (test (< ?lit ?cap))
=> (modify ?j (litros ?cap)))

(defrule vaciarjarra ;Vacia jarra
?j <- (JARRA (nombre ?n) (litros ?lit))
 (test (> ?lit 0))
=> (modify ?j (litros 0)))

```

(defrule volcarjarra ; Vacia el contenido de una jarra en la otra
?j1 <- (JARRA (nombre ?n1) (capacidad ?cap1) (litros ?lit1))
?j2 <- (JARRA (nombre ?n2) (capacidad ?cap2) (litros ?lit2))
(test (and (neq ?n1 ?n2) (> ?lit1 0) (<= (+ ?lit1 ?lit2) ?cap2)))
=> (modify ?j1 (litros 0))
(modify ?j2 (litros (+ ?lit2 ?lit1)))))

(defrule verterjarra ; Vierto una jarra hasta llenar otra
?j1 <- (JARRA (nombre ?n1) (capacidad ?cap1) (litros ?lit1))
?j2 <- (JARRA (nombre ?n2) (capacidad ?cap2) (litros ?lit2))
(test (and (neq ?n1 ?n2) (< ?lit2 ?cap2) (> (+ ?lit1 ?lit2) ?cap2)))
=> (modify ?j1 (litros (- ?lit1 (- ?cap2 ?lit2)))))
(modify ?j2 (litros ?cap2)))

(defrule acabar ;Comprueba terminación
(declare (salience 1000)) ;regla con máxima prioridad para terminar
?j <- (JARRA (litros 2))
=> (printout t "Solución encontrada " crlf)
(halt)) ;si se encuentra una solución se detiene el razonamiento

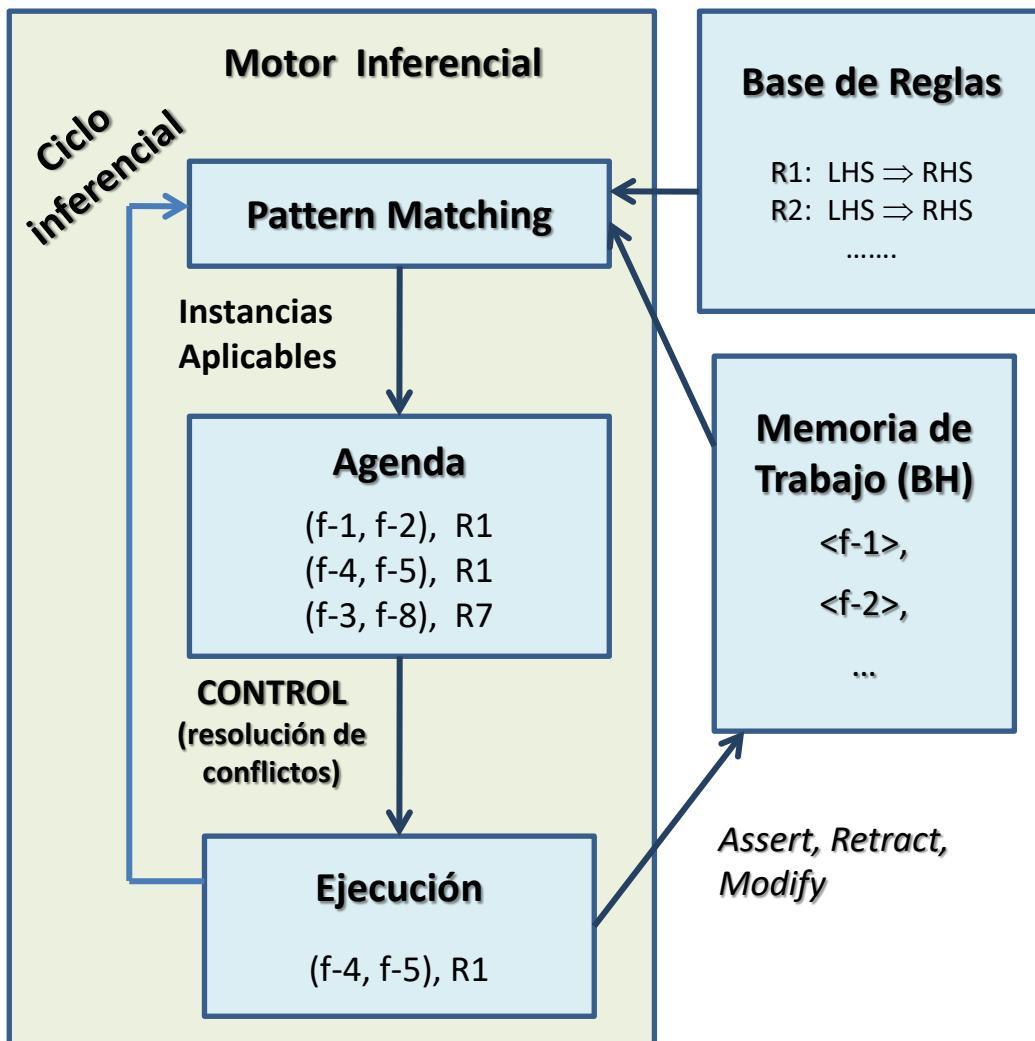
```

¿Siempre se encontrará la solución?

La ejecución de este sistema dará fácilmente lugar a un ciclo!!!

1.2.- Razonamiento. Inferencia y Control.

El **proceso inferencial** utiliza el conocimiento sobre el **dominio** (hechos y reglas), y sobre el **problema** (hechos), para construir (buscar) la **línea de razonamiento** que conduce a la **solución o respuesta** al problema.



Conceptos del Motor de Inferencia:

• Encadenamiento Inferencial

(Qué obtener):

- Encadenamiento hacia adelante
- Encadenamiento hacia atrás

• Control Inferencial

(Cómo obtenerlo):

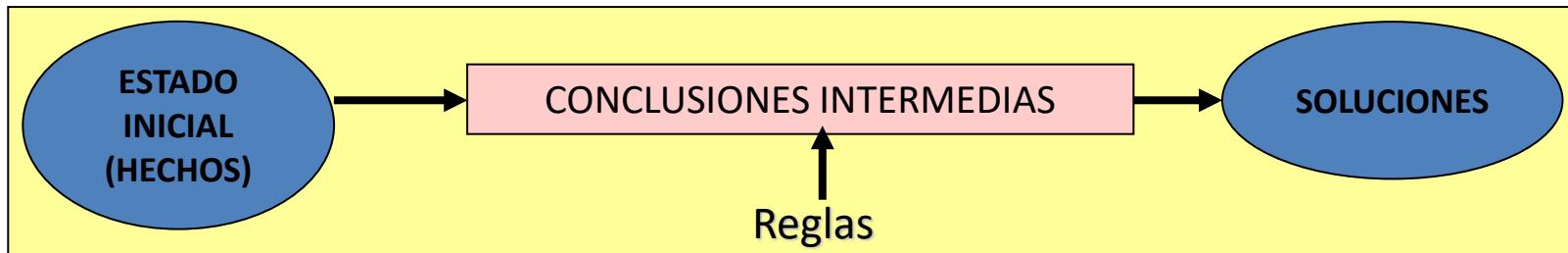
- Anchura, Profundidad, etc.
- Estrategias de Control

Encadenamiento Inferencial

Encadenamiento Hacia-Adelante: Dirigido por los datos (forward)

Implicaciones usadas como F-reglas que operan sobre la BH inicial hasta satisfacer los objetivos.

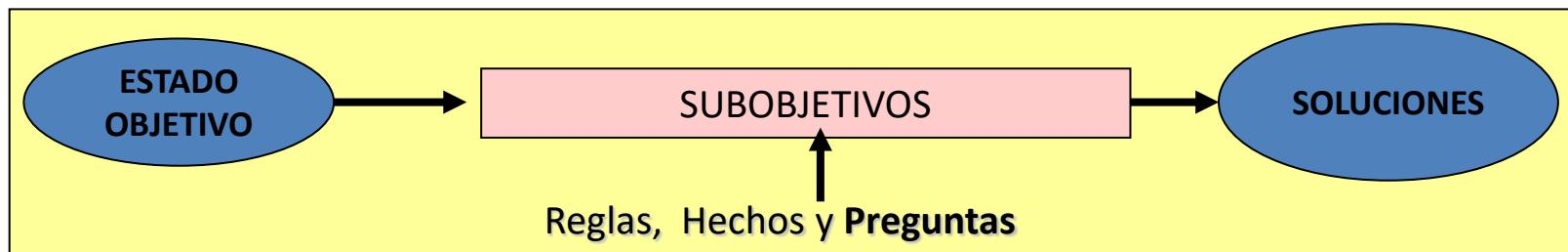
(se deduce todo lo que se puede deducir)



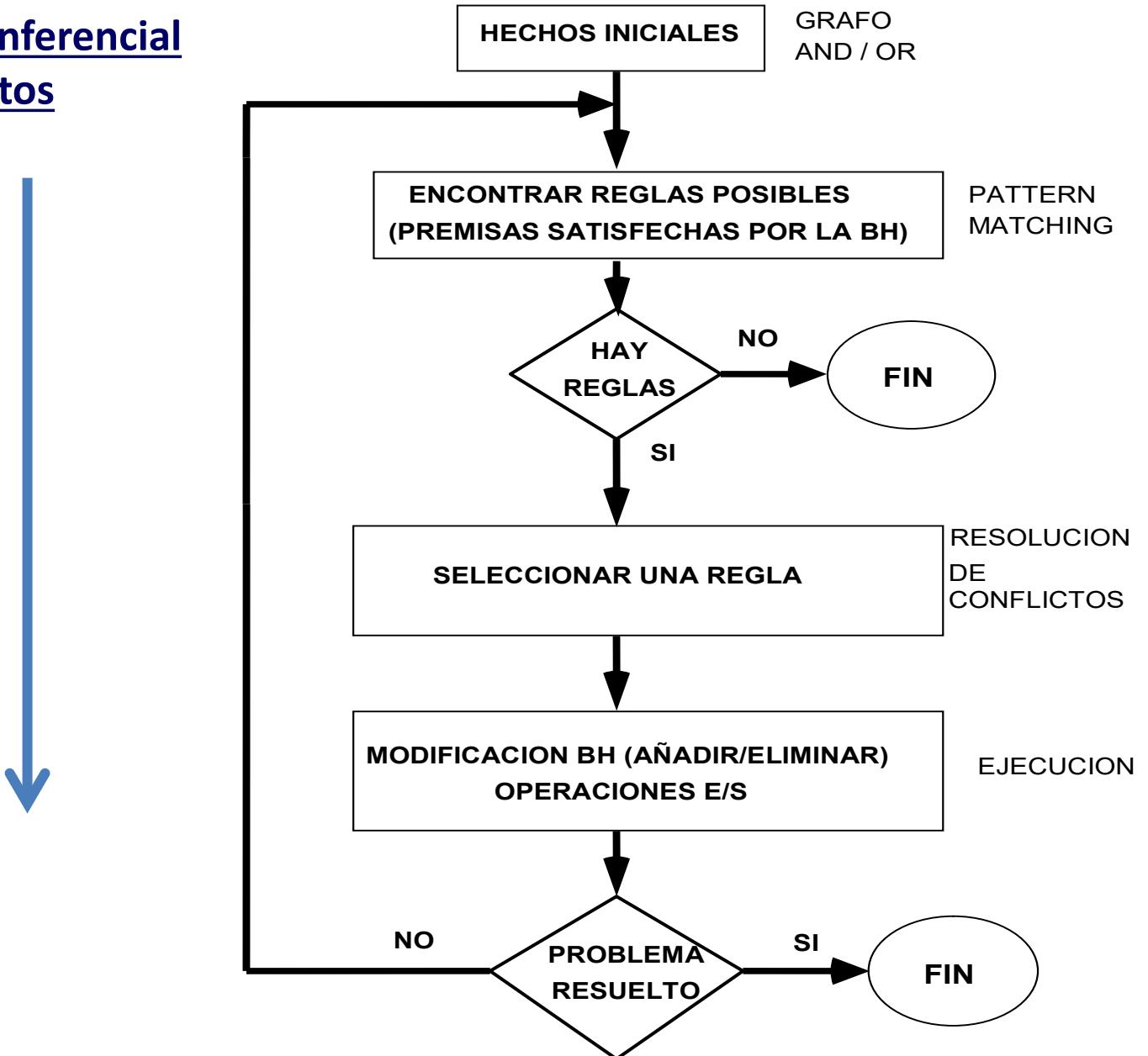
Encadenamiento Hacia-Atrás: Dirigido por los objetivos (backward)

Implicaciones usadas como B-reglas, partiendo del objetivo, que van obteniendo subobjetivos parciales hasta que se satisface la condición de terminación representada por la BH inicial.

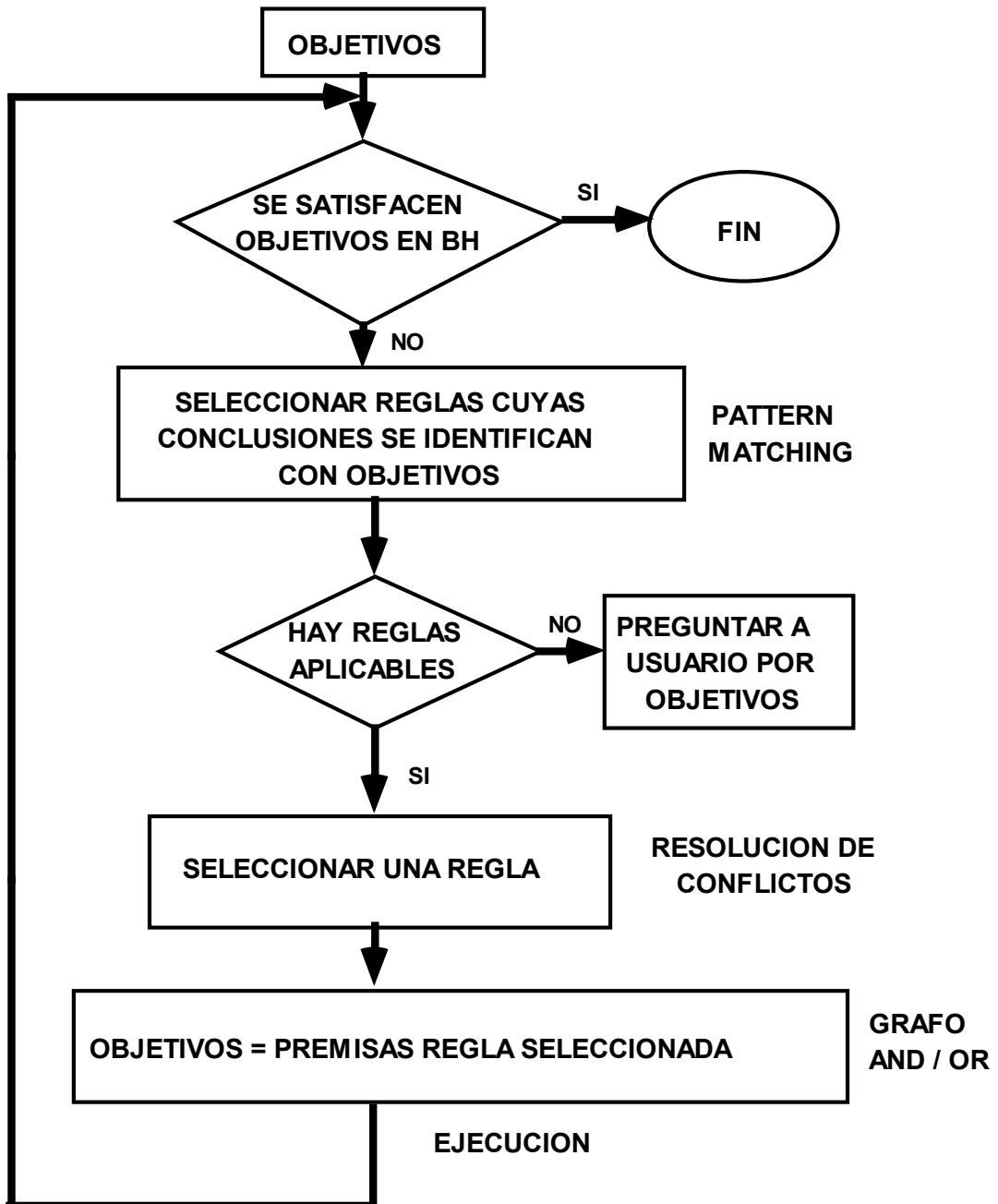
(se deduce solo lo que es necesario en la demostración)



Encadenamiento Inferencial dirigido por los datos



Encadenamiento inferencial dirigido por el objetivo



Ejemplo Encadenamiento hacia delante

HECHOS INICIALES:

EL COCHE NO ARRANCA
EL DEPOSITO NO ESTA VACIO
LAS LUCES FUNCIONAN

REGLA BATERIA:

SI LAS LUCES FUNCIONAN
ENTONCES LA BATERIA ESTA BIEN
(ASERCIÓN)

REGLA STARTER:

SI EL COCHE NO ARRANCA
Y LA BATERIA ESTA BIEN
Y EL DEPOSITO NO ESTA VACIO
ENTONCES
HAY UN PROBLEMA EN EL STARTER
(ASERCIÓN)

CONCLUSIONES:

LA BATERIA ESTA BIEN
HAY UN PROBLEMA EN EL STARTER

Ejemplo Encadenamiento hacia atrás

HECHO INICIAL: EL COCHE NO ARRANCA

Hipótesis a resolver:

¿EL COCHE TIENE UN PROBLEMA EN EL STARTER?

USANDO LA REGLA DEL STARTER:

SI EL COCHE NO ARRANCA (*hecho conocido*)
Y LA BATERIA ESTA BIEN (*regla batería*)
Y EL DEPOSITO NO ESTA VACIO
(*preguntado al usuario y asertado*)

ENTONCES HAY UN PROBLEMA DE STARTER

USANDO LA REGLA BATERIA:

SI LAS LUCES FUNCIONAN
(*preguntado al usuario y asertado*)
ENTONCES LA BATERIA ESTA BIEN

CONCLUSION:

HAY UN PROBLEMA DE STARTER
(*demonstrado verdadero y asertado*)

Características Hacia Adelante	Características Hacia Atrás
<p>Procedimiento Razona-adelante (BH: BH inicial, S: Reglas)</p> <p>Mientras $S \neq \emptyset$ y problema no-resuelto hacer:</p> <p style="padding-left: 2em;"><i>Definir</i> S (instancias de reglas aplicables).</p> <p style="padding-left: 2em;"><i>Control:</i> Seleccionar R de S.</p> <p style="padding-left: 2em;"><i>Aplicar</i> R, redefiniendo BH</p>	<p>Procedimiento Razona-atrás (G: Objetivo, BH conocida, S: Reglas)</p> <p>Definir S (instancias de reglas con consecuentes en G)</p> <p>Si S está vacío, <i>preguntar al usuario</i></p> <p>Si no:</p> <p style="padding-left: 2em;">Mientras G sin definir y S no vacío, hacer:</p> <p style="padding-left: 2em;"><i>Control:</i> Seleccionar R de S.</p> <p style="padding-left: 2em;">Considerar los antecedentes A de R.</p> <p style="padding-left: 2em;">Si $\exists A' \subset A$ no definido: Razona-atrás (A') sino: Definir G.</p>
<ul style="list-style-type: none"> • Guiado por los Datos. • Sistemas generativos: Obtiene todos los hechos deducibles. • Control no importante si: <ul style="list-style-type: none"> Monótono (no retracta) Todas las acciones son relevantes 	<ul style="list-style-type: none"> • Guiado por los objetivos. Más parecido a la conducta humana (eficacia). • Necesita <i>objetivos explícitos</i>, que intenta cumplir. • Obtiene respuestas a preguntas. • Permiten preguntar al usuario. • Control importante: Orden preguntas, satisfacción subobjetivos (eficacia)
<p>Aplicaciones:</p> <p>Sistemas que no precisan delimitar respuesta. Todas, o una respuesta posible.</p> <p>Configuración, Control, Predicción, Simulación, Monitorización.</p>	<p>Aplicaciones:</p> <p>Sistema con Respuesta Definida.</p> <p>Consultivos, Clasificativos, De Interpretación, Diagnóstico, Planificación.</p>

Ejercicio Resolver el siguiente problema en base a las siguientes reglas

Reglas:

R1: A → C

R2: A → H

R3: C → D

R4: D → E

R5: B ∧ F → X

Encadenamiento hacia adelante

Hecho inicial: A

¿Qué hechos se generan en un razonamiento forward?

¿Se genera siempre la misma información?

Encadenamiento hacia atrás

Hecho inicial: A

Hecho objetivo: E

¿Qué hechos se generan en un razonamiento backward? ¿Se resuelve el objetivo?

¿Coincide la información generada en encadenamiento hacia adelante y hacia atrás?

Control Inferencial

Selección de la instancia de regla a aplicar de entre las posibles:

Control por Agenda de Instancias de Reglas

Control inferencial:

Amplitud, Profundidad,

Complejidad de premisas (más específica)

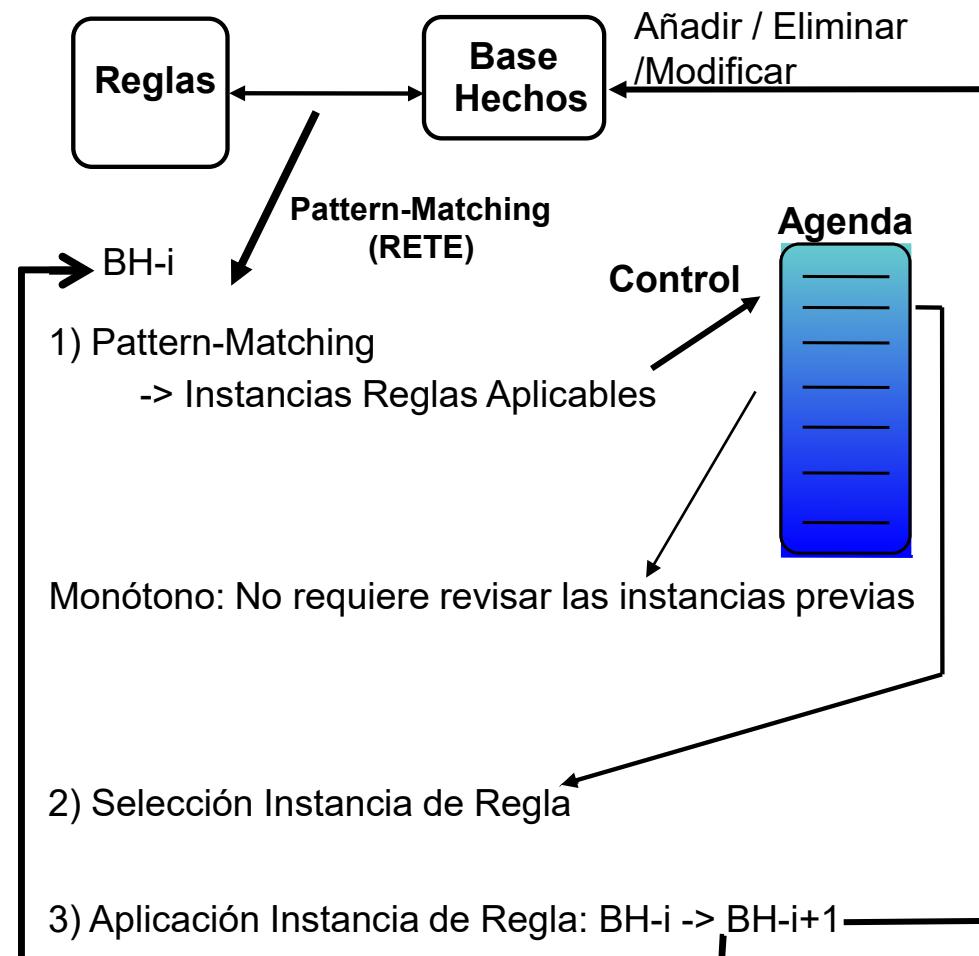
Peso de las reglas,

Combinación: peso, complejidad

Mayor frecuencia

Última utilizada

- **Metarreglas (Cómo utilizar el conocimiento):**
 - Cambio encadenamiento
 - Clases de Reglas:
 - Utilización de reglas.
 - Exclusión de reglas.
 - Asignación dinámica de pesos / Prioridades



Encadenamiento Forward y Control

El control inferencial en el razonamiento hacia adelante no es relevante

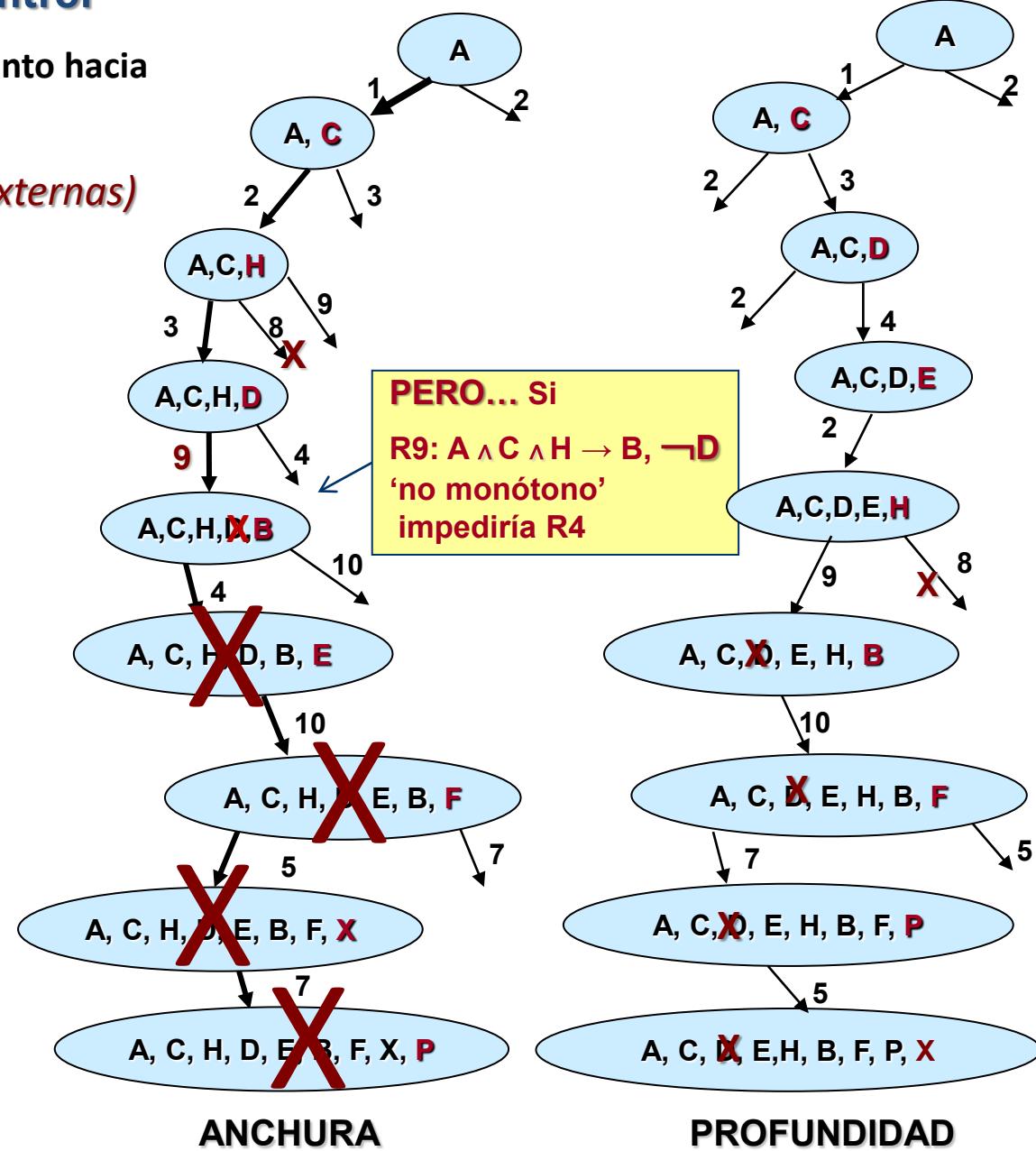
(salvo retractaciones o acciones externas)

Reglas:

- R1: $A \rightarrow C$
- R2: $A \rightarrow H$
- R3: $C \rightarrow D$
- R4: $D \rightarrow E$
- R5: $B \wedge F \rightarrow X$
- R6: $D \wedge G \rightarrow B$
- R7: $C \wedge F \rightarrow P$
- R8: $A \wedge H \rightarrow C$
- R9: $A \wedge C \wedge H \rightarrow B$
- R10: $A \wedge B \wedge E \wedge H \rightarrow F$

Estado-inicial: {A}

mismas conclusiones...



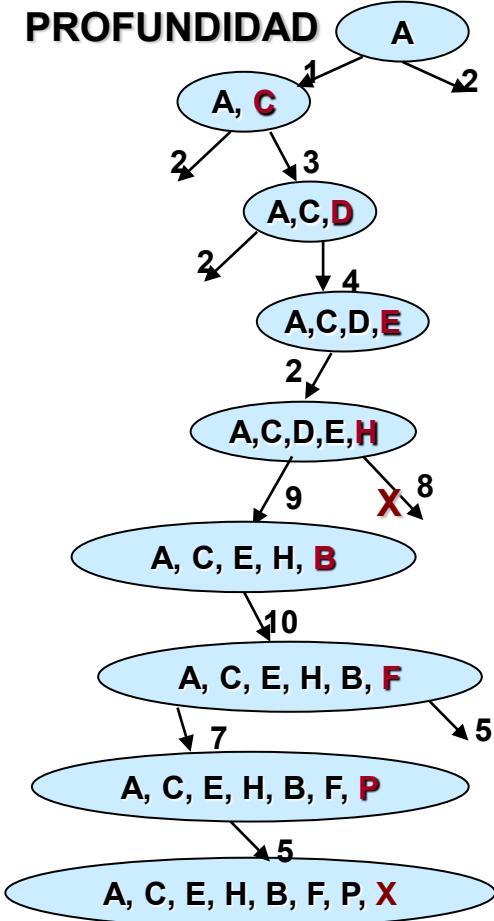
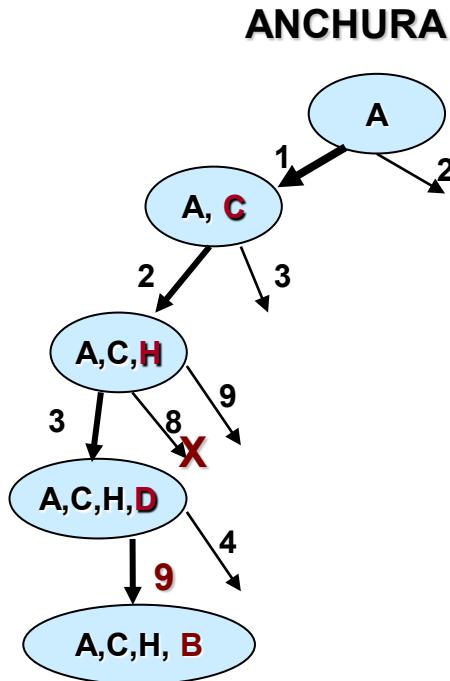
Encadenamiento Forward y Control (caso no monótono)

Reglas:

- R1: $A \rightarrow C$
- R2: $A \rightarrow H$
- R3: $C \rightarrow D$
- R4: $D \rightarrow E$
- R5: $B \wedge F \rightarrow X$
- R6: $D \wedge G \rightarrow B$
- R7: $C \wedge F \rightarrow P$
- R8: $A \wedge H \rightarrow C$

R9: $A \wedge C \wedge H \rightarrow B, \neg D$

R10: $A \wedge B \wedge E \wedge H \rightarrow F$



En un sistema no-monótono (*las condiciones que se satisfacen pueden disminuir conforme se avanza*), se pueden obtener distintas conclusiones según el control:

- Nueva información puede retractar hechos previamente obtenidos.
- Se hacen hipótesis que pueden ser invalidadas conforme se contrastan las suposiciones, etc.

*Afeitarse eléctricamente, no es peligroso.
Ducharse, no es peligroso.
Afeitarse eléctricamente y Ducharse, es peligroso.*

*Las aves vuelan, Tweety es un ave, Tweety vuela.
Tweety es un pingüino,
Tweety no vuela.*

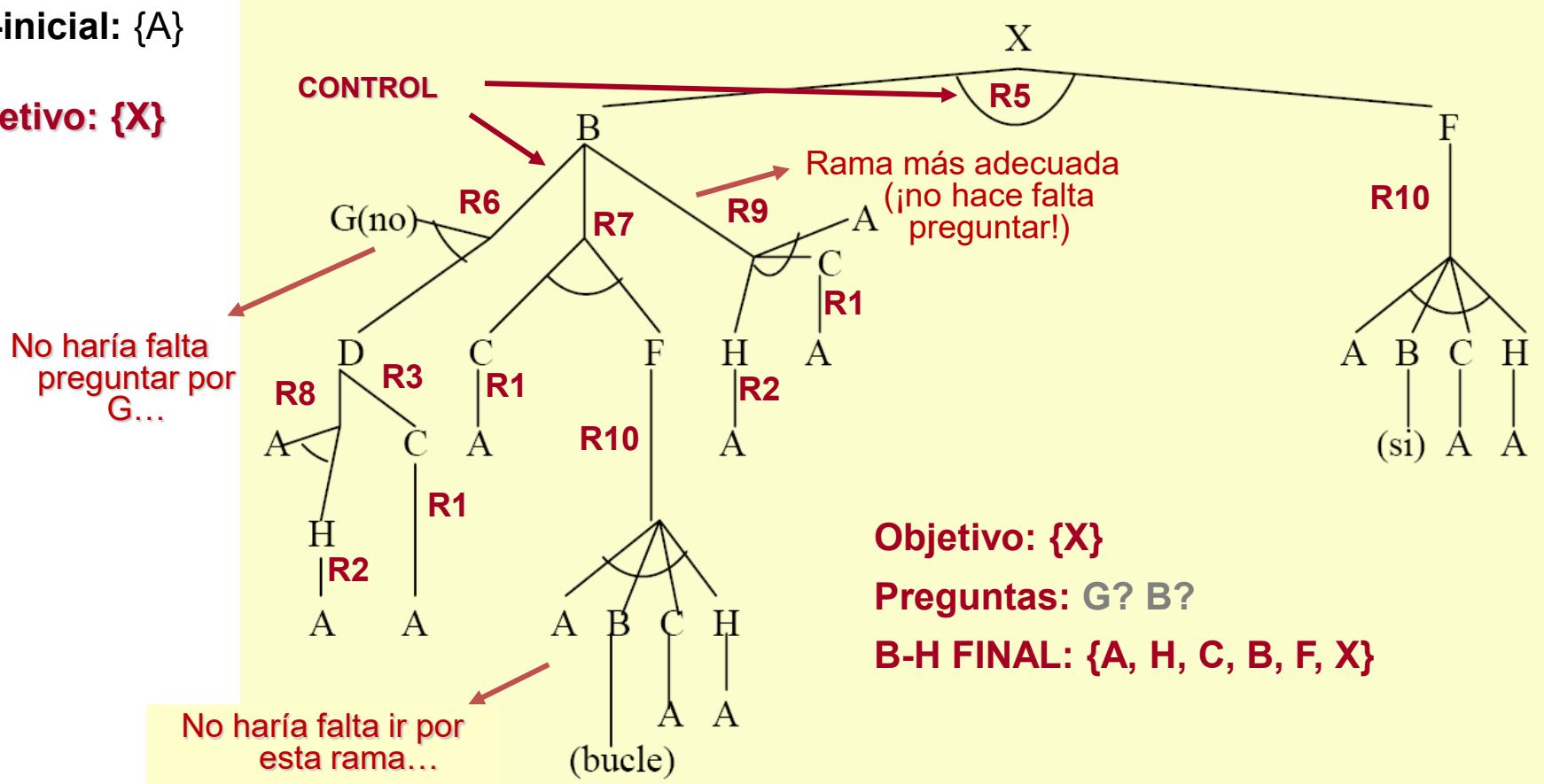
Encadenamiento Backward y Control

El control inferencial en el **razonamiento hacia atrás** determina la **senda** para alcanzar el objetivo (y la secuencia de **preguntas** que se harán al usuario): **fiabilidad, rapidez, etc.**

Reglas: R1: $A \rightarrow C$ R6: $D \wedge G \rightarrow B$ R2: $A \rightarrow H$ R7: $C \wedge F \rightarrow B$
R3: $C \rightarrow D$ R8: $A \wedge H \rightarrow D$ R4: $D \rightarrow E$ R9: $A \wedge C \wedge H \rightarrow B$ R5: $B \wedge F \rightarrow X$
R10: $A \wedge B \wedge C \wedge H \rightarrow F$

BH-inicial: {A}

Objetivo: {X}



Encadenamiento Inferencial Backward

R1: si Animal:Cubre=pelo entonces Animal:Tipo=mamifero.

R2 si Animal:Cubre=plumas y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.

R3 si Animal:Tipo=mamifero y Animal:Come=carne y Animal:Garras=sí entonces Animal:Alimenta=carnívoro.

R4 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Piel=manchas entonces Animal:Nombre=guepardo.

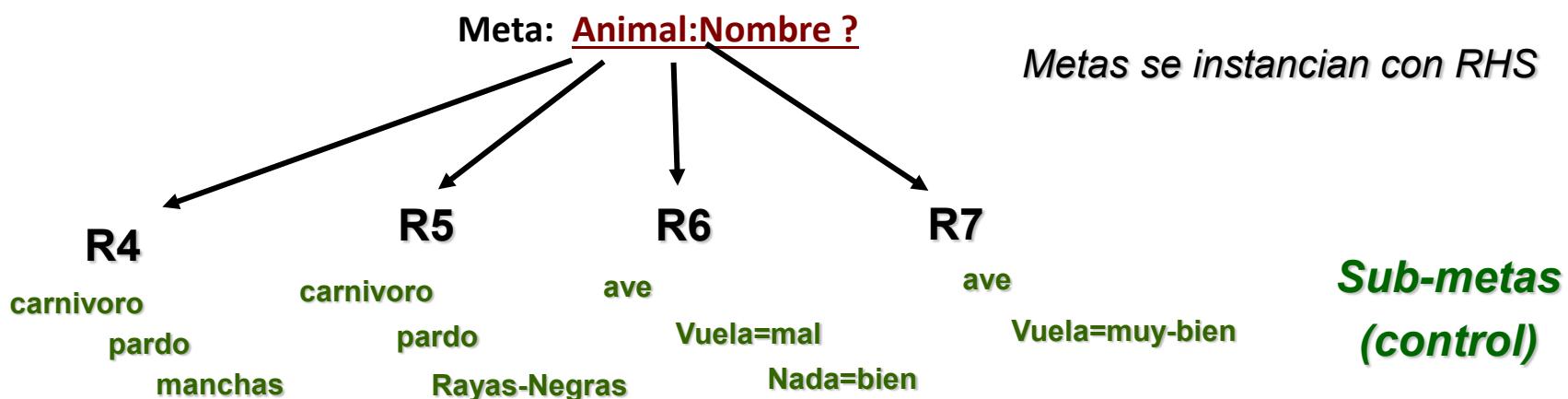
R5 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Rayas=negras entonces Animal:Nombre=tigre.

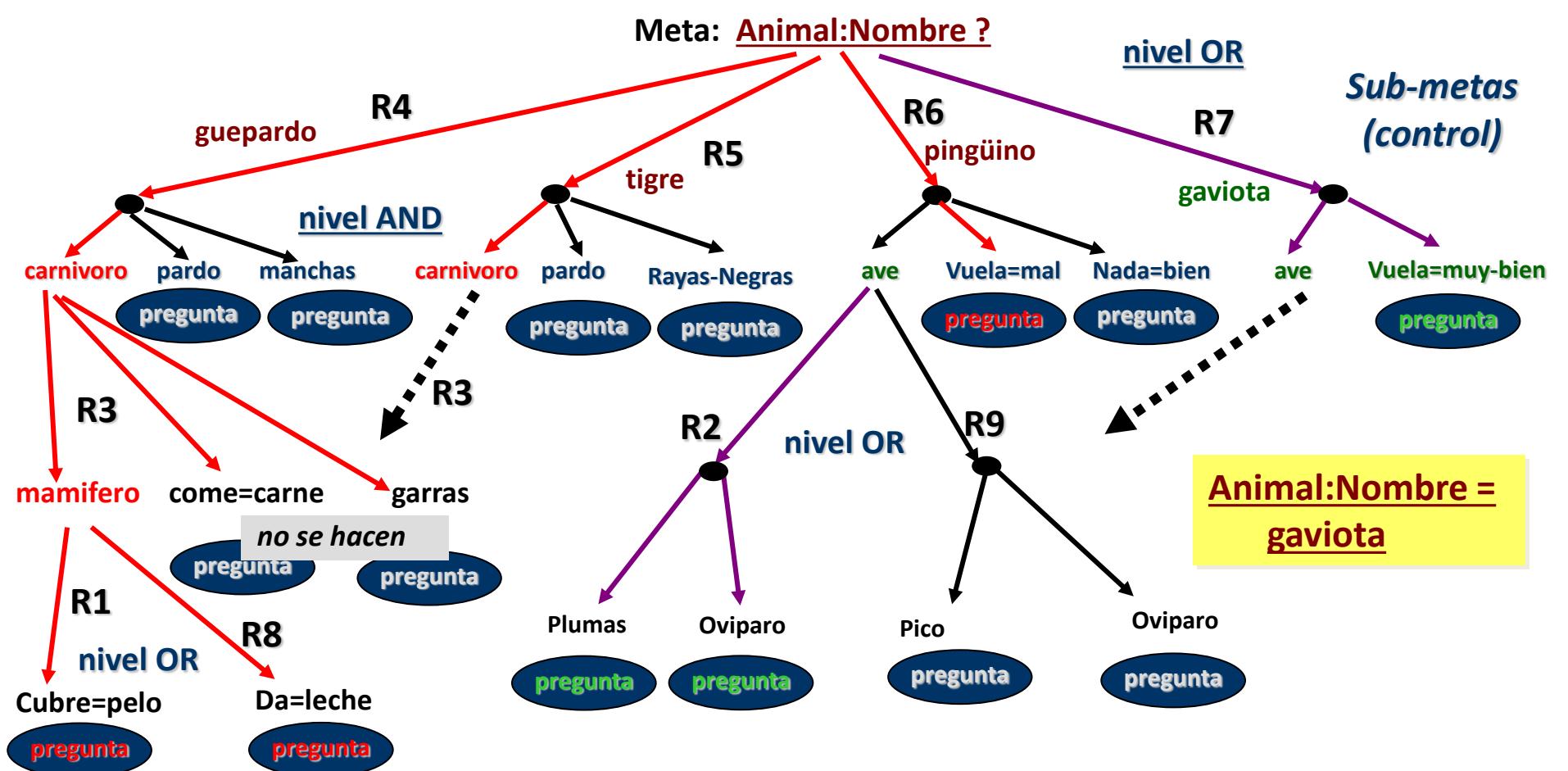
R6 si Animal:Tipo=ave y Animal:Vuela=mal y Animal:Nada=bien Entonces Animal:Nombre=pingüino.

R7 si Animal:Tipo=ave y Animal:Vuela=muy bien entonces Animal:Nombre=gaviota.

R8 si Animal:Da leche=sí entonces Animal:Tipo=mamifero.

R9 si Animal:Pico=sí y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.





R1: si Animal:Cubre=pelo entonces Animal:Tipo=mamifero.

R2 si Animal:Cubre=plumas y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.

R3 si Animal:Tipo=mamifero y Animal:Come=carne y Animal:Garras=sí entonces Animal:Alimenta=carnívoro.

R4 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Piel=manchas entonces Animal:Nombre=guepardo.

R5 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Rayas=negras entonces Animal:Nombre=tigre.

R6 si Animal:Tipo=ave y Animal:Vuela=mal y Animal:Nada=bien Entonces Animal:Nombre=pingüino.

R7 si Animal:Tipo=ave y Animal:Vuela=muy_bien entonces Animal:Nombre=gaviota.

R8 si Animal:Da_leche=sí entonces Animal:Tipo=mamifero.

R9 si Animal:Pico=sí y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.

Cubre-pelo?
Da leche?
Plumas?
Ovíparo?
Vuela mal?
Vuela muy bien?
⇒ AVE, gaviota

Ejercicio Generar el árbol and-or con las preguntas correspondientes para Meta: Animal:Nombre ?

Animal:Nombre =
tigre

R1: si Animal:Cubre=pelo entonces Animal:Tipo=mamifero.

R2 si Animal:Cubre=plumas y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.

R3 si Animal:Tipo=mamifero y Animal:Come=carne y Animal:Garras=sí entonces Animal:Alimenta=carnívoro.

R4 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Piel=manchas entonces Animal:Nombre=guepardo.

R5 si Animal:Alimenta=carnívoro y Animal:Color=pardo y Animal:Rayas=negras entonces Animal:Nombre=tigre.

R6 si Animal:Tipo=ave y Animal:Vuela=mal y Animal:Nada=bien Entonces Animal:Nombre=pingüino.

R7 si Animal:Tipo=ave y Animal:Vuela=muy bien entonces Animal:Nombre=gaviota.

R8 si Animal:Da leche=sí entonces Animal:Tipo=mamifero.

R9 si Animal:Pico=sí y Animal:Reproduce=Oviparo entonces Animal:Tipo=ave.

- **Encadenamiento inferencial:**

- Tipos: forward (guiado por datos), backward (guiado por objetivos) o híbrido (bidireccional)
- Dependiente del tipo de aplicación (generativa o clasificativa).
- Forward y backward NO son equivalentes:

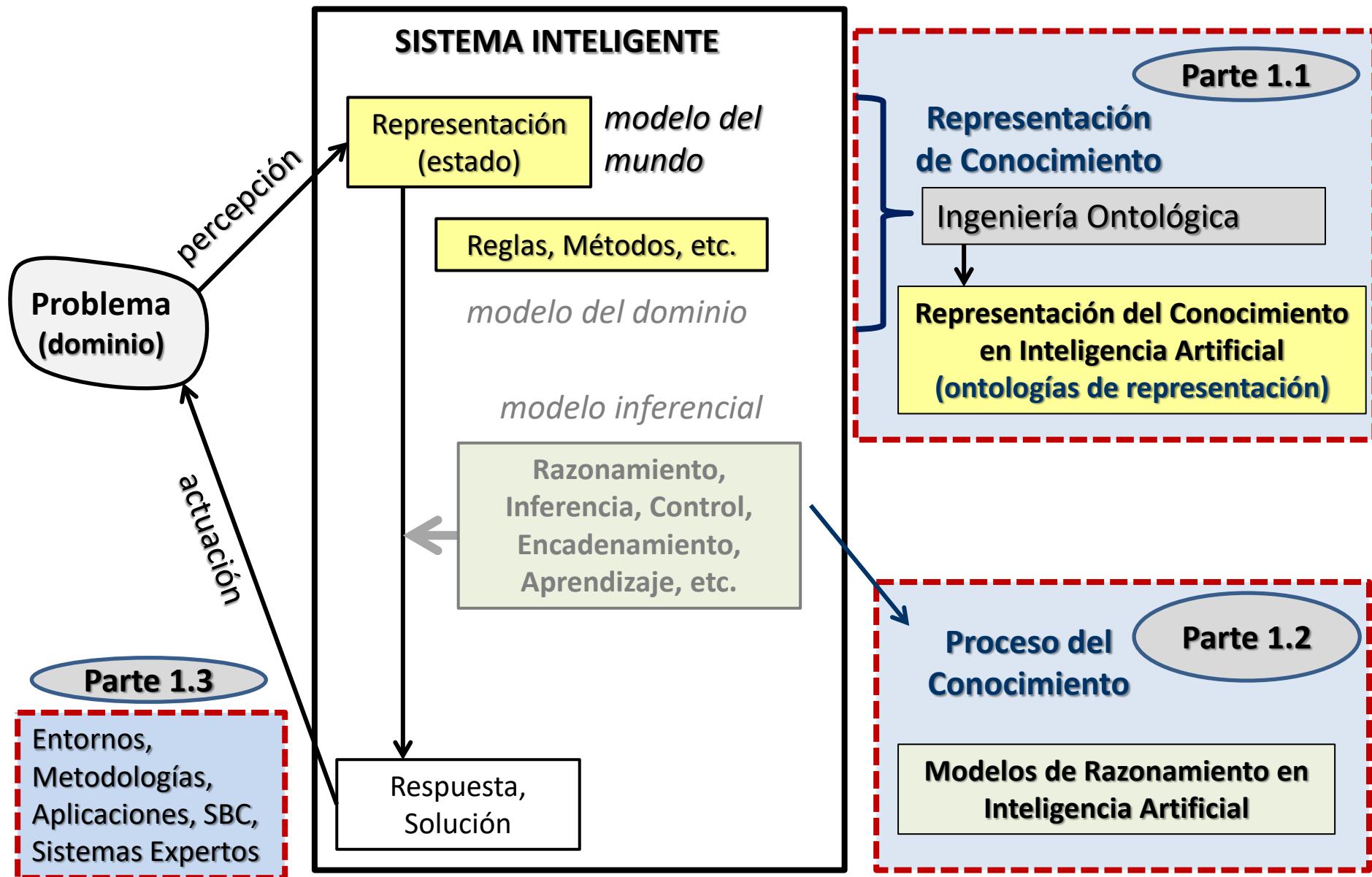
Forward suele generar más información que backward.

Backward se centra en la información más relevante para el objetivo, permite preguntar al usuario y, en *general*, suele ser más eficiente

- **Control Inferencial** (selección de reglas en conflicto):

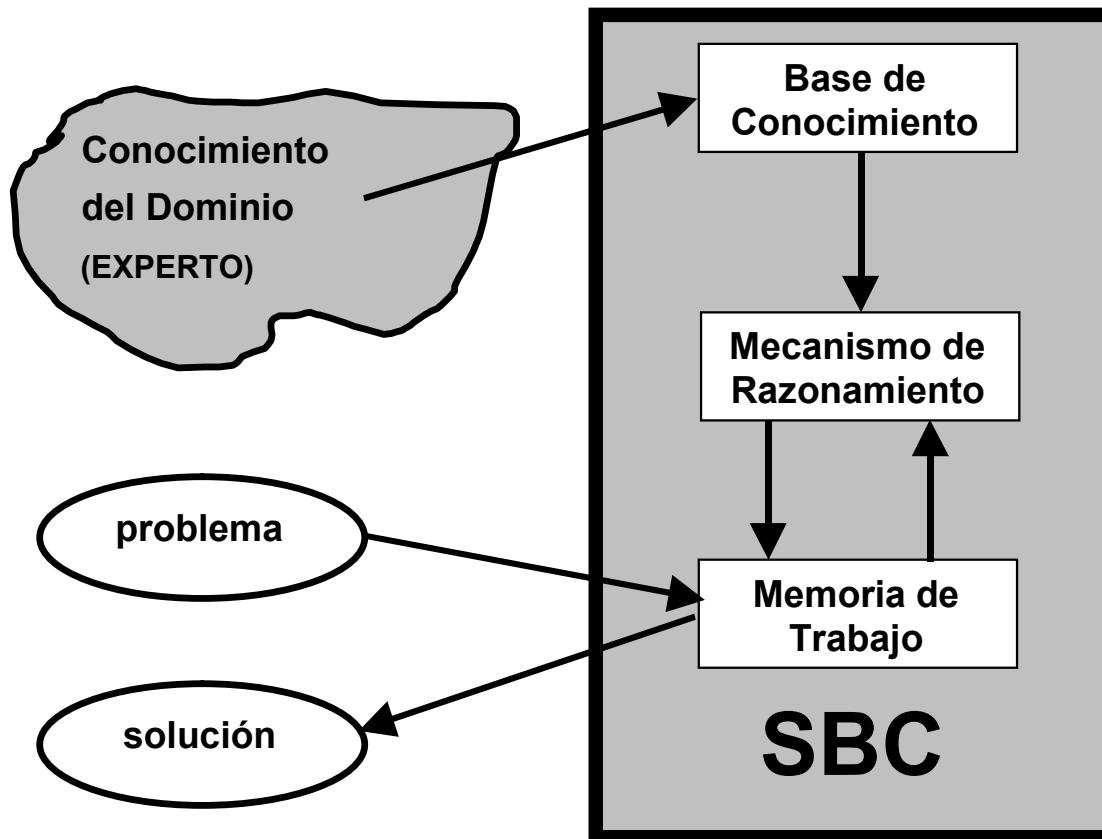
- Tipos: Anchura, profundidad, prioridad (salience) o personalizado
- El mecanismo de selección de reglas NO debe ser arbitrario
- En algunos casos pueden generar la misma información (pero en distinto orden), pero en otros casos puede diferir (particularmente, si se retracta/elimina información)

1.3.- Sistemas Basados en el Conocimiento, Sistemas Expertos. Aplicaciones.



¿Qué es un Sistema Experto?

Un Sistema Experto es un SBC, cuyo conocimiento se obtiene a partir de la **habilidad de un experto humano**, de forma que el sistema puede dar consejos o tomar decisiones inteligentes con una competencia similar, y es capaz de justificar/explicar sus respuestas.

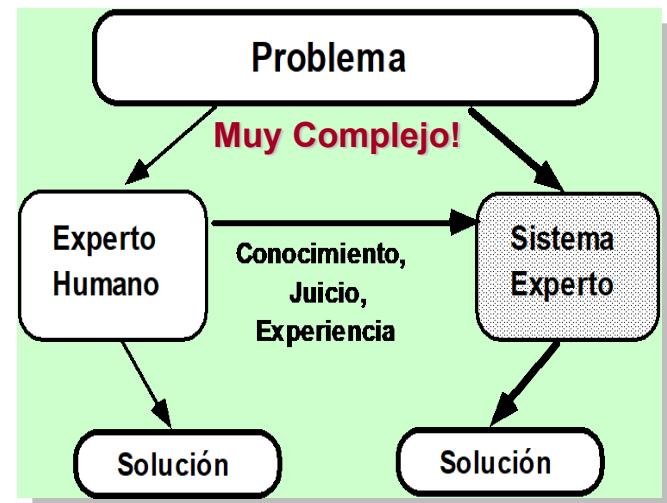


El experto humano...

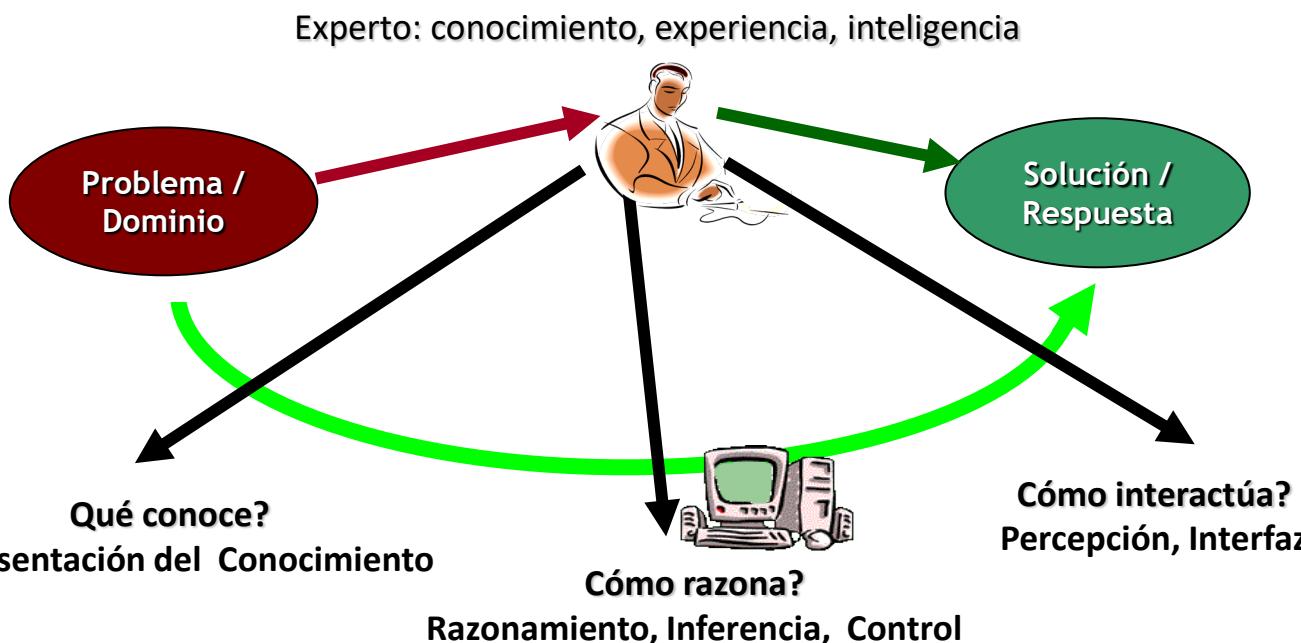
- conoce el dominio de aplicación y el problema a resolver.
- es razonable y tiene experiencia.
- aplica un esquema de razonamiento adecuado al problema.

Un S.E. razona utilizando:

- **conocimiento experto** para resolver problemas complejos de un **dominio**
- conocimiento del **problema concreto**.
- aplicando técnicas de razonamiento (heurísticas, inciertas, difusas, hipotéticas, etc.)



Además es capaz de **justificar y explicar** su línea de razonamiento, preguntas y conclusiones.



Un SE **emula** la capacidad de un experto humano en la resolución de problemas difíciles.

Aplicaciones típicas de los Sistemas Expertos

- **Gestión empresarial y finanzas:**

- Soporte de decisión
- Promoción y selección de personal
- Análisis estado de la organización
- Estrategias tácticas
- Estrategias de negociación
- Análisis de operaciones crediticias
- Análisis de mercados
- Análisis financieros y contables
- Análisis y gestión de riesgos
- Planificación de inversiones
- Asesoría fiscal.

- **Educación**

- Sistemas tutores

- **Leyes:**

- Asesoría legal.
- Análisis aplicación leyes

Geología:

- Ayuda prospecciones geológicas
- Int. datos prospecciones geológicas
- Cálculo de la superficie geológica

- **Química:**

- Cálculo de estructuras moleculares
- Identificación de componentes

- **Supervisión y diagnóstico:**

- Centrales eléctricas
- Sistemas electrónicos
- Máquinas herramienta
- Análisis de vibraciones
- Tráfico urbano y aéreo
- Otros procesos industriales
- Análisis de seguridad.

- **Operación y control**

- Ferrocarriles
- Sistemas termoeléctricos
- Centrales nucleares
- Refinerías de petróleo
- Industrias cerámicas
- Cementeras
- Centrales eléctricas
- Redes telefónicas

- **Planificación:**

- Maniobras en sistemas eléctricos.
- Empresas de transporte aéreo.
- Empresas de transporte terrestre
- Empresas de transporte marítimo
- Robots

Defensa:

- Planificación estratégica / táctica.
- Interpretación de señales.
- Control y guiado vehículos.

- **Medicina. Diagnóstico y Tratamiento**

- Enfermedades infecciosas
- Enfermedades pulmonares
- Glaucoma
- Neumonía
- Medicina interna
- Nefrología.
- Enfermedades oncológicas

- **Ayuda al diseño:**

- Máquinas eléctricas
- Sistemas espaciales
- Circuitos electrónicos
- Computadores
- Sistemas termoeléctricos

- **Insigth, OPS5, ART, ART-IM:** Entornos iniciales de desarrollo.
- **KEE, Goldworks :** Lisp , potente, flexible, frames/objetos, interfaz. (80-90's).
- **KappaPC:** versión PC de KEE (90's). Frames, backward, interfaz.
- **FLEX / VisiRule: (90's)** PC-win (prolog), flexible, frames, no interfaz usuario.
- **GURU/G2:** Orientado a tiempo real (*control y aplicaciones críticas*). No backward (*event-driven*).
- **ACQUIRE:** Especial adquisición conocimiento mediante pattern-maching de casos.
- **Nexpert Object:** PC, flexible, frames, interfaz. Comercial.
- **Jess (~clips en Java).** Libre.
- **Protégé.** Orientado al modelado. No razonamiento (Jess / Clips).
- **Otros:**

EZ-Xpert (<http://www.ez-xpert.com/>),

Amzi! (<http://www.amzi.com/>),

EXSYS (<http://www.exsys.com/>), etc.

Determinación de Meta-clases (Ing. Conocimiento):

Información de Médico:

Se observa una Temperatura de 41ºC, lo que anormalmente es fiebre alta y síntoma de una infección bacterial.

→ [Metaclases Observables](#)

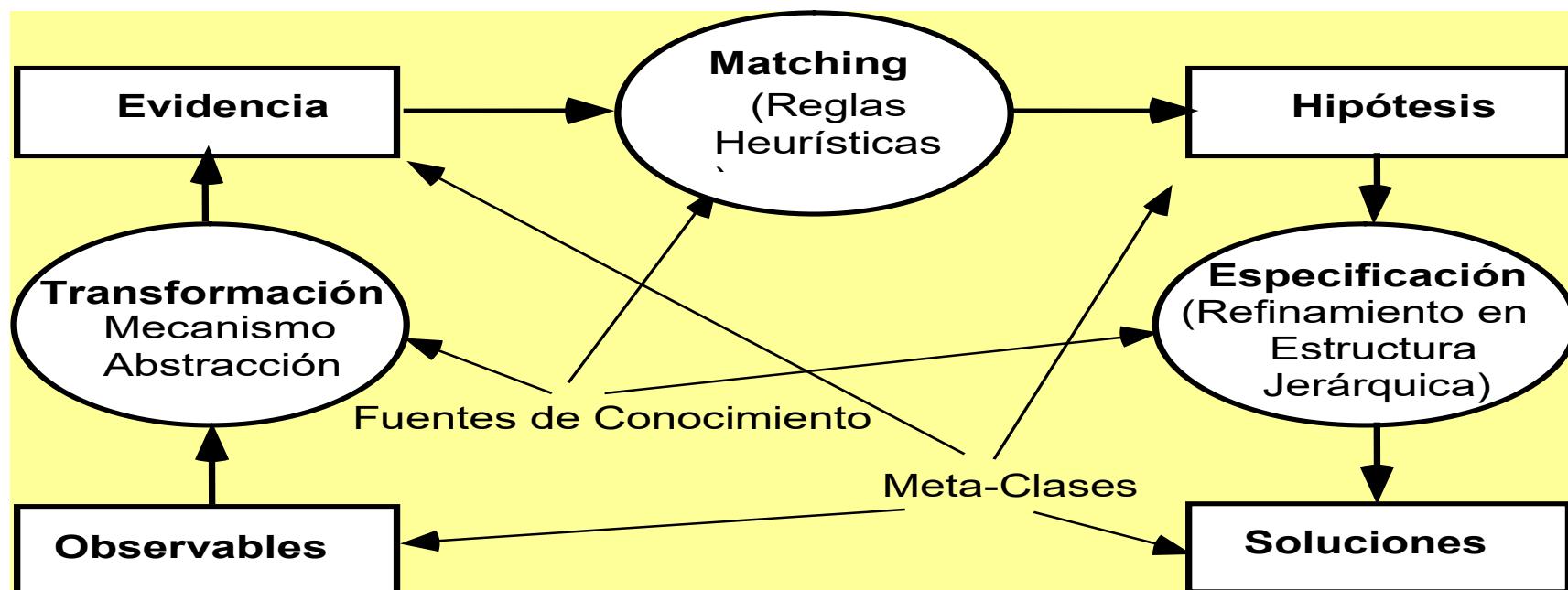
Al medir la Presión diastólica se observa que es 110mmHg. Por ello, se concluye que el paciente padece una Neumonía causada por pneumococcea.

→ [Metaclases Evidencias](#)

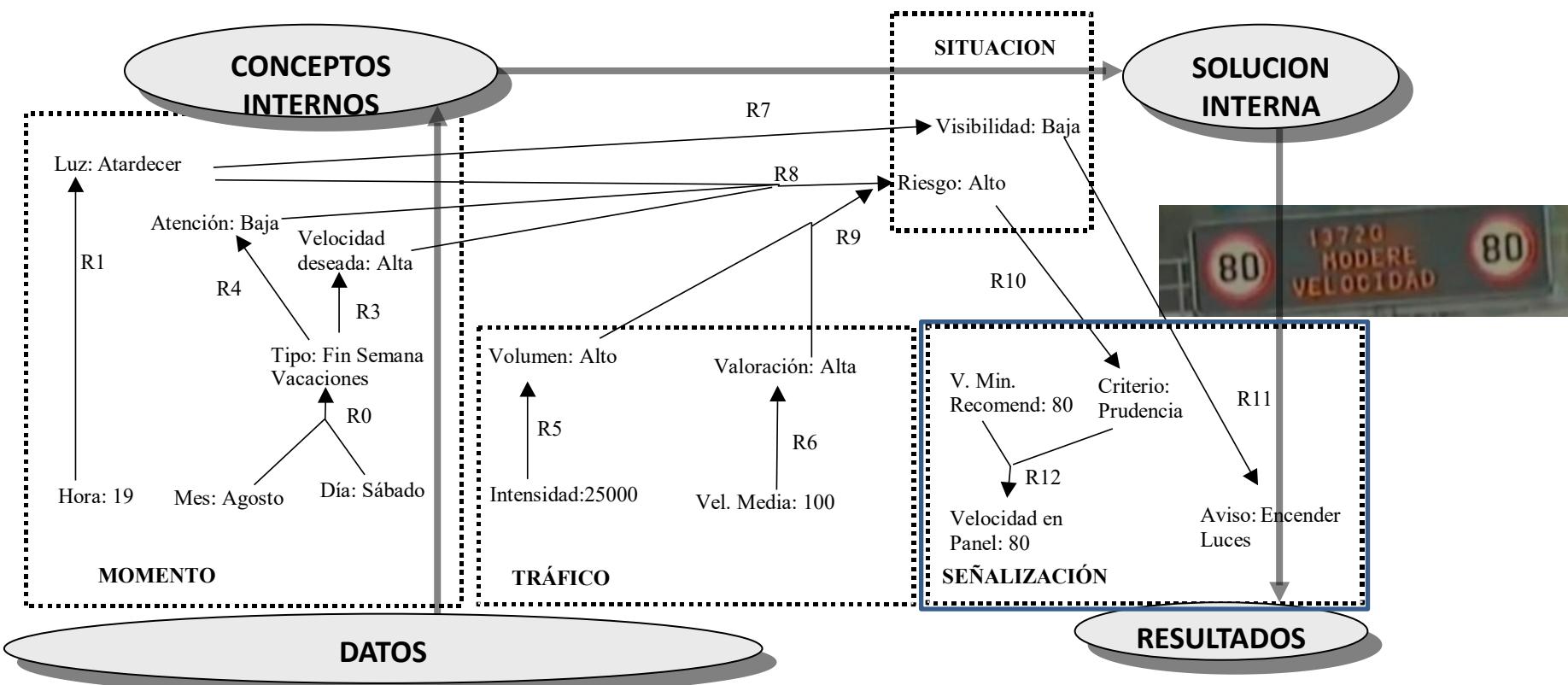
→ [Metaclases Hipótesis](#)

→ [Metaclases Soluciones](#)

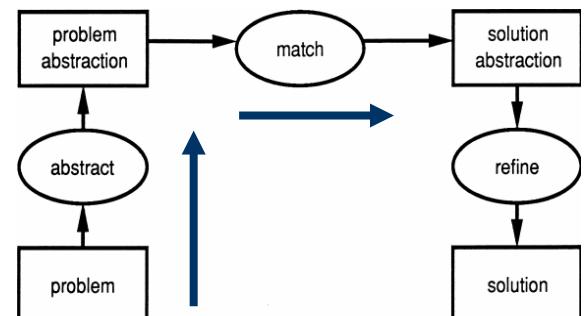
Estructura Inferencial:



EJEMPLO MODELO INFERENCIAL: INDICACIONES EN UN PANEL DE TRÁFICO



Clasificación Heurística



- Los **Sistemas Expertos** constituyen una de las más típicas y clásicas aplicaciones de la IA.
- Un Sistema Experto combina **información del dominio** y del **problema** para, utilizando dicho conocimiento, **razonar** (incorporando y simulando lo que haría un experto humano) y así ofrecer una **solución justificada de competencia similar**.
- El **campo de aplicaciones** de los SE es muy amplio: diagnóstico, recomendación, diseño, control, planificación, depuración, monitorización, etc.
- Existen diversas **metodologías de desarrollo** de SE (inspiradas en metodologías de Ing. SW).
 - Una metodología particularmente aceptada: **Common-Kads**: basado en capas interrelacionadas con un conjunto de plantillas textuales.
 - Principal problema: **adquisición conocimiento** y modelado inferencial
- Como ocurre en Ing. SW, es importante llevar a cabo el proceso de **Verificación y Validación** del SE para evaluar su bondad (corrección, utilidad, robustez, rendimiento, etc.)

Realiza un SBC para modelar un sencillo sistema de diagnóstico de un equipo de audio formado por varios elementos como alimentación, reproductor, altavoces, etc. A priori, cualquiera de estos elementos puede funcionar mal.

Un sistema de audio consta de amplificador, cables, reproductor de CD y altavoces, cualquiera de los cuales puede funcionar mal. De todas las posibles explicaciones del mal funcionamiento, me centro primero en las que tienen un origen eléctrico. Esto se detecta fácilmente por las luces de los displays de los dos aparatos. Si alguna está apagada, se ha comprobado el origen eléctrico del problema que normalmente se debe a un olvido de pulsar ON en el interruptor ON/OFF. Menos habitual es un olvido de conectar el enchufe.

Si estas explicaciones no son válidas puede ser que haya un fallo de alimentación. Para ello compruebo si hay corriente en la red eléctrica. Es menos probable que la causa sea una rotura del cable, lo que es observable a primera vista. Si las luces están encendidas, entonces compruebo que en el amplificador se ha seleccionado el CD como fuente (el display muestra "CD"), que hay un CD en el reproductor y que está en modo de reproducción (su display muestra "Play").

Si ninguna de estas causas se confirma, la avería puede deberse a la rotura de alguno de los cables de sonido. Para comprobar si el amplificador está averiado conecto cualquier otra fuente aparte del reproductor de CD's. La última posibilidad es que esté averiado el reproductor.