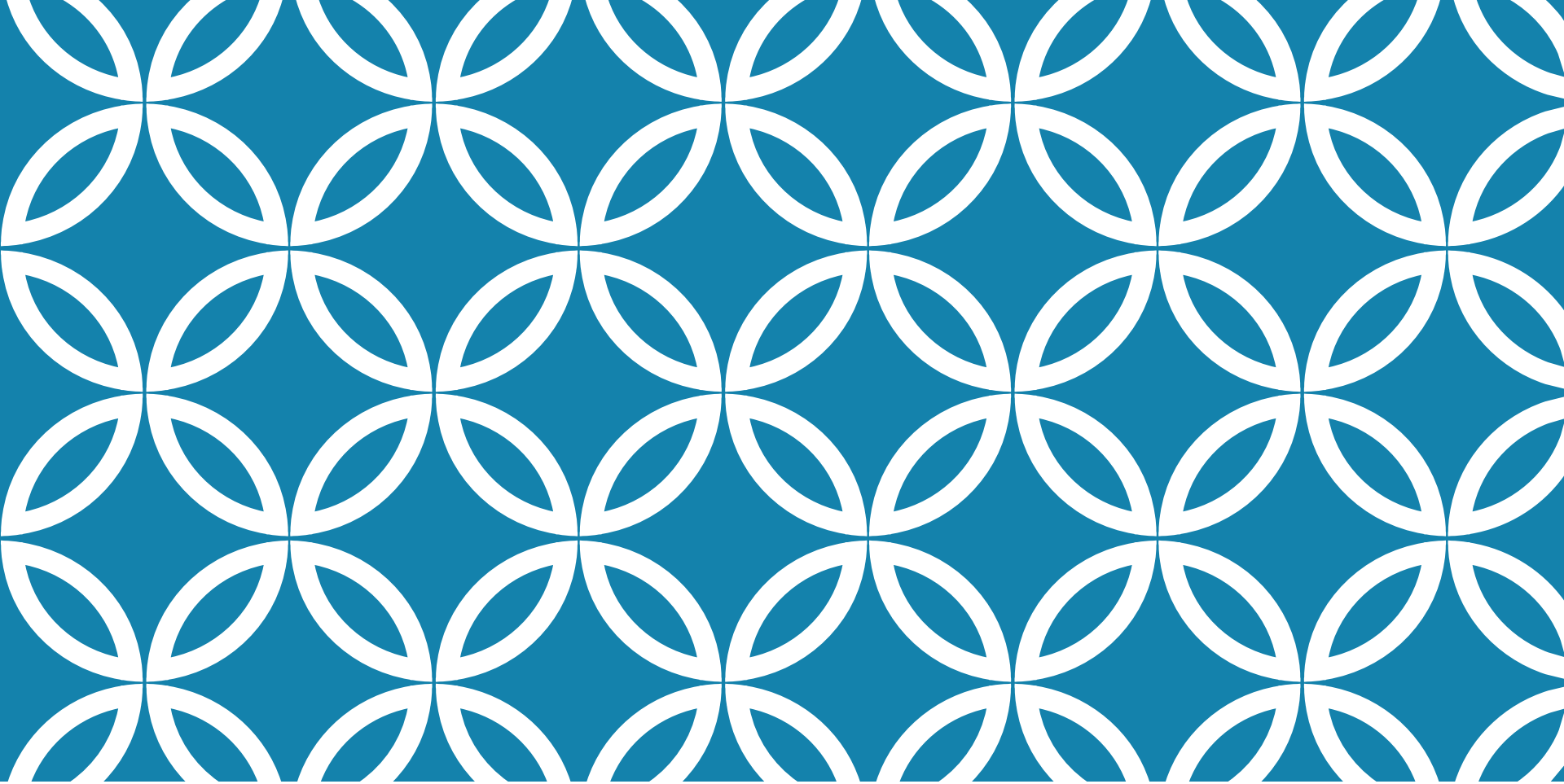


REPRESENTACIÓN DEL CONOCIMIENTO

José Manuel Molina López
Grupo de Inteligencia Artificial Aplicada

CONTENIDO

Representación I. Metodologías y Técnicas

- Ingeniería del Conocimiento
- Revisión Histórica
- Ontologías

Representación II. Sistemas de producción

- Introducción a los Sistemas de Producción
- Representación del Conocimiento en un Sistema de Producción
  - Hechos
  - Reglas
- Ciclo de ejecución
  - Equiparación y activación
  - Resolución de conflictos
  - Ejecución
- Ejercicios
- Aplicaciones
  - Características
  - Aplicaciones

Revisión Histórica Ingeniería del Conocimiento

Ontologías

METODOLOGÍAS Y TÉCNICAS

REVISIÓN HISTÓRICA

Sistemas de reglas
Marcos
Incertidumbre

SISTEMAS DE PRODUCCIÓN

Una regla no es más que la transcripción del conocimiento de una persona a un formato lógico.

Cadena de Inferencia: reglas que están interconectadas pues las conclusiones de unas son premisas para otras.

Intérprete : idenfica reglas aplicables y selecciona el orden en el que serán ejecutadas. Dos tipos:

Forward chaining o dirigido por los datos.	Backward chaining o dirigido por el resultado.

PROGRAMACIÓN LÓGICA CON RESTRICCIONES

Un conjunto de atributos con un dominio de valores posibles finito.

Un conjunto de restricciones que condicionan los valores que pueden tomar los atributos anteriores.

Filtrado de hipótesis que asignan combinaciones de valores a los atributos anteriores.

Razonamiento temporal: puntual, intervalos, restricciones temporales.

MARCOS O 'FRAMES'

Humanos agrupamos conocimiento en marcos conceptuales (Minsky, 1975). Tratar nuevas situaciones a partir del parecido con situaciones anteriores.

Marco o Frame con atributos valorados en un rango, con posible valor por defecto.

El valor de un atributo puede calcularse mediante procedimientos asociados (demonios) que se ejecutan cuando:

- Se necesita. Se añade un determinado valor. Se cambia un determinado valor.

Jerarquía de marcos permite representar especialización y generalización.

Equiparación de marco pregunta con el que más encaje.

REDES SEMÁNTICAS

(Quillian,1968) para representar el conocimiento en frases.

Representan desde la semántica hasta relaciones causales.

Grafo dirigido etiquetado. Etiquetas de nodos son nombres (conceptos), de los arcos son verbos (relaciones).

Inferencia mediante herencia y equiparación de patrones.

Ventajas : Explícitas y sucintas. Reducen el tiempo de búsqueda.

Desventajas : No hay interpretaciones estándar. Se producen inferencias inválidas. Explosión combinatoria.

RAZONAMIENTO CON INCERTIDUMBRE

¿Cómo tratar con conceptos que van más allá de las etiquetas “verdadero” y “falso”?.

Las reglas de inferencia se basan en impresiones de expertos que no son capaces de afirmar la veracidad o falsedad.

Definir cómo se representa la incertidumbre, como se calcula y como se propaga por las reglas

Teorema de bayes: probabilidades

Factores de certidumbre

Teoría de Demster-Shafer

Conjuntos borrosos y lógica difusa

INGENIERÍA DEL CONOCIMIENTO

Conocimiento

Metodología (SSBBCC-Ingeniería del Conocimiento)

SISTEMAS BASADOS EN EL CONOCIMIENTO

La ingeniería de conocimiento produce SBC.

SBC: Sistema que usa conocimiento específico del dominio del problema.

Sistema Experto: Tareas que requieren razonamiento humano.

Que pretende comportarse como un experto.

Conocimiento representado explícitamente de forma separada (Base de Conocimientos).

Funcionamiento no algorítmico, incluye heurísticas

SISTEMAS BASADOS EN EL CONOCIMIENTO

REPRESENTACIÓN

¡¡¡¡Conocimiento representado explícitamente de forma separada (Base de Conocimientos)!!!!

BÚSQUEDA

¡¡¡Funcionamiento no algorítmico, incluye heurísticas!!!

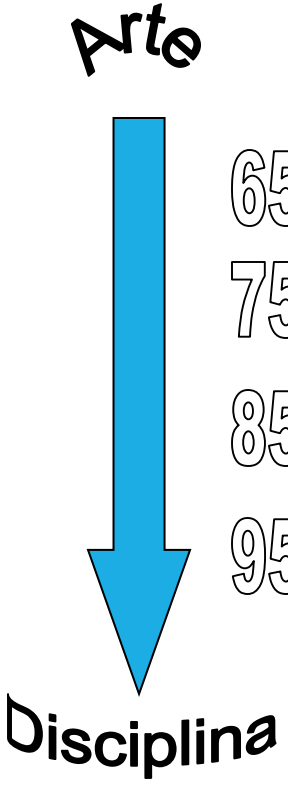
INGENIERÍA DE CONOCIMIENTO

Proceso de **adquirir, estructurar, formalizar y hacer operativos** un conjunto de conocimientos en un programa (SBC) que resuelva una tarea compleja adecuadamente.

Importante por:

- Conocimiento tiene valor por sí mismo y sobrevive a implementaciones.
- Los errores en el conocimiento → decisivos
- Facilita escalabilidad y mantenimiento.

PRECEDENTES Y EVOLUCIÓN



65 Sistemas de propósito general (GPS)

75 Sistemas basados en reglas (Mycin)

85 Adolescencia de metodología (KADS)

95 Madurez de metodología (CommonKADS)

HISTORIA DE KADS

Una década de esfuerzo (1983-1993).

1982 necesidad de proceso de desarrollo de SBC más estructurado.

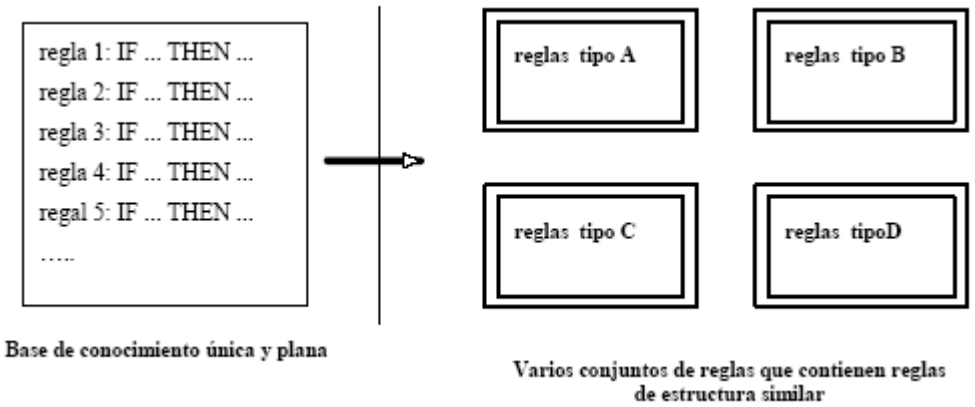
Proyecto ESPRIT en 1983 centrado en adquisición de conocimiento (análisis de técnicas de educción).

Proyecto KADS-I, 1985: Teoría, herramientas, validación basada en casos.

Proyecto KADS-II, 1990: Modelo social, todo el ciclo de vida, dotar de una especificación formal del modelo de conocimiento.

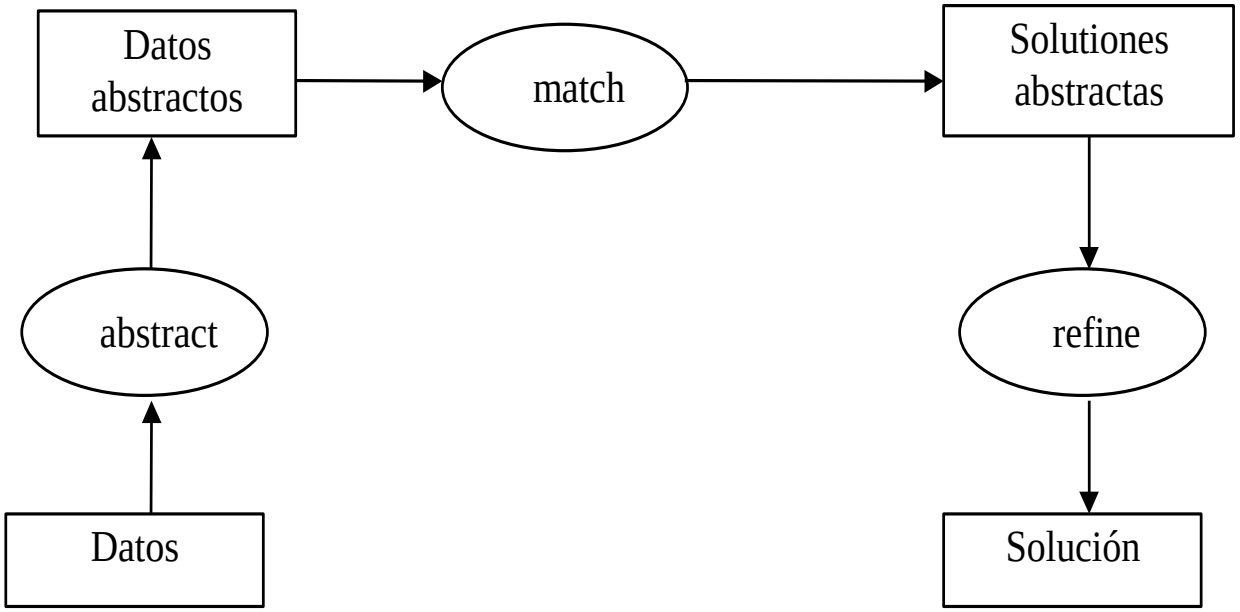
Common-Kads: completa metodología para el desarrollo de SS.BB.CC. Estándar de facto.

PRECEDENTES A KADS



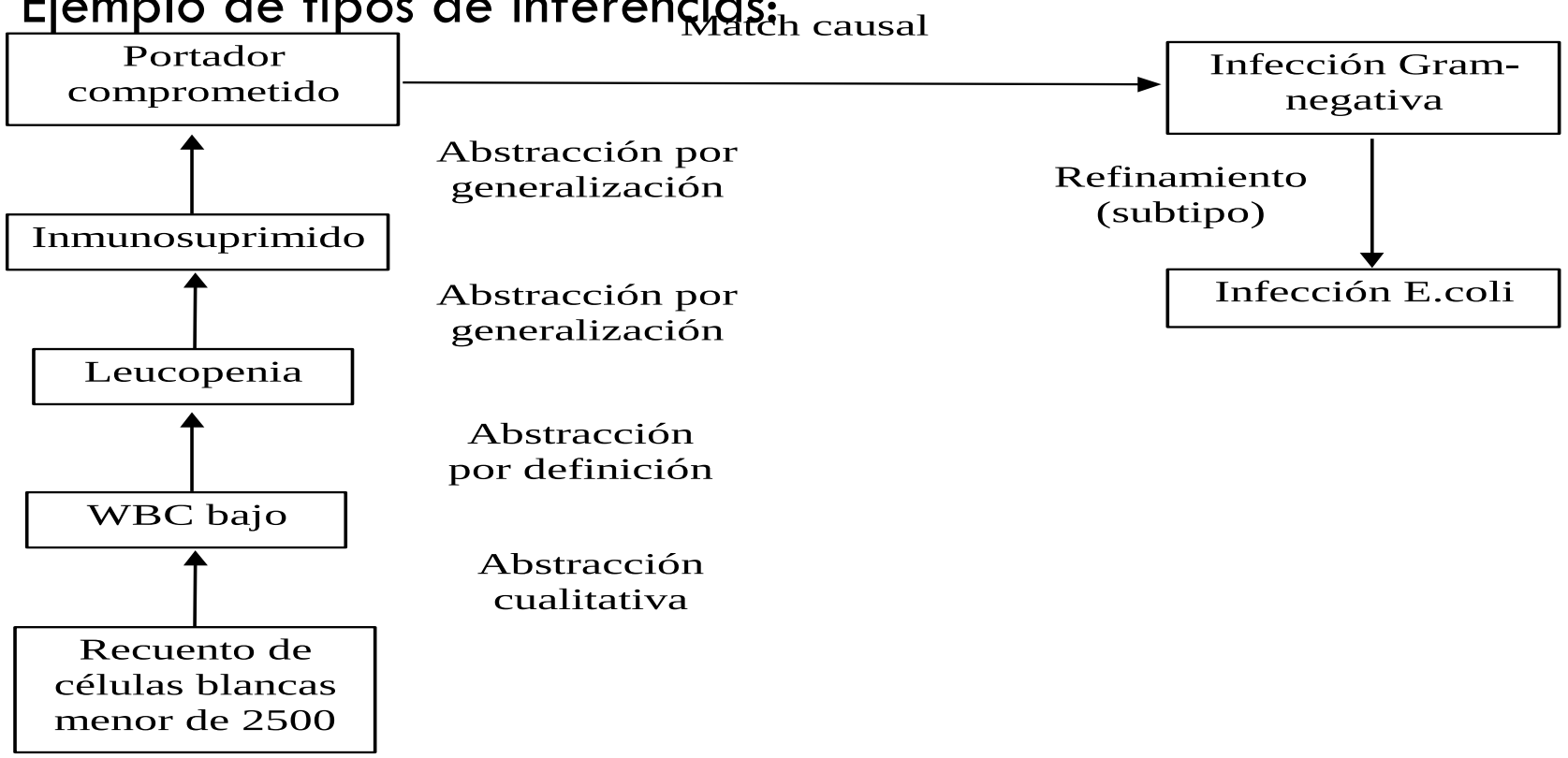
PRECEDENTES A KADS

Tres tipos de inferencias: Abstracción, Relación, Refinamiento (Clancey, 1985).



PRECEDENTES DE KADS

Ejemplo de tipos de inferencias:



PRECEDENTES DE KADS

Cuatro tipos de tareas genéricas: clasificación, interpretación, diagnosis y construcción.

Tareas específicas son instancias de tareas genéricas: comparten elementos significativos en común, mismo método de resolución del problema y los mismos modelos del dominio.

PRINCIPIOS DE KADS

Actividad de modelización parcial del conocimiento humano. Un modelo es una abstracción de una parte de la realidad con un propósito específico.

El conocimiento debe modelarse atendiendo a los conceptos e ignorando a la posible futura implementación.

El conocimiento siempre tiene una estructura interna estable que se puede analizar categorías y patrones semejantes en diferentes problemas.

Permite mejoras estructuradas a partir de modelos intermedios. Gestión del proyecto flexible, pero más controlada que prototipado.

MODELOS DE KADS

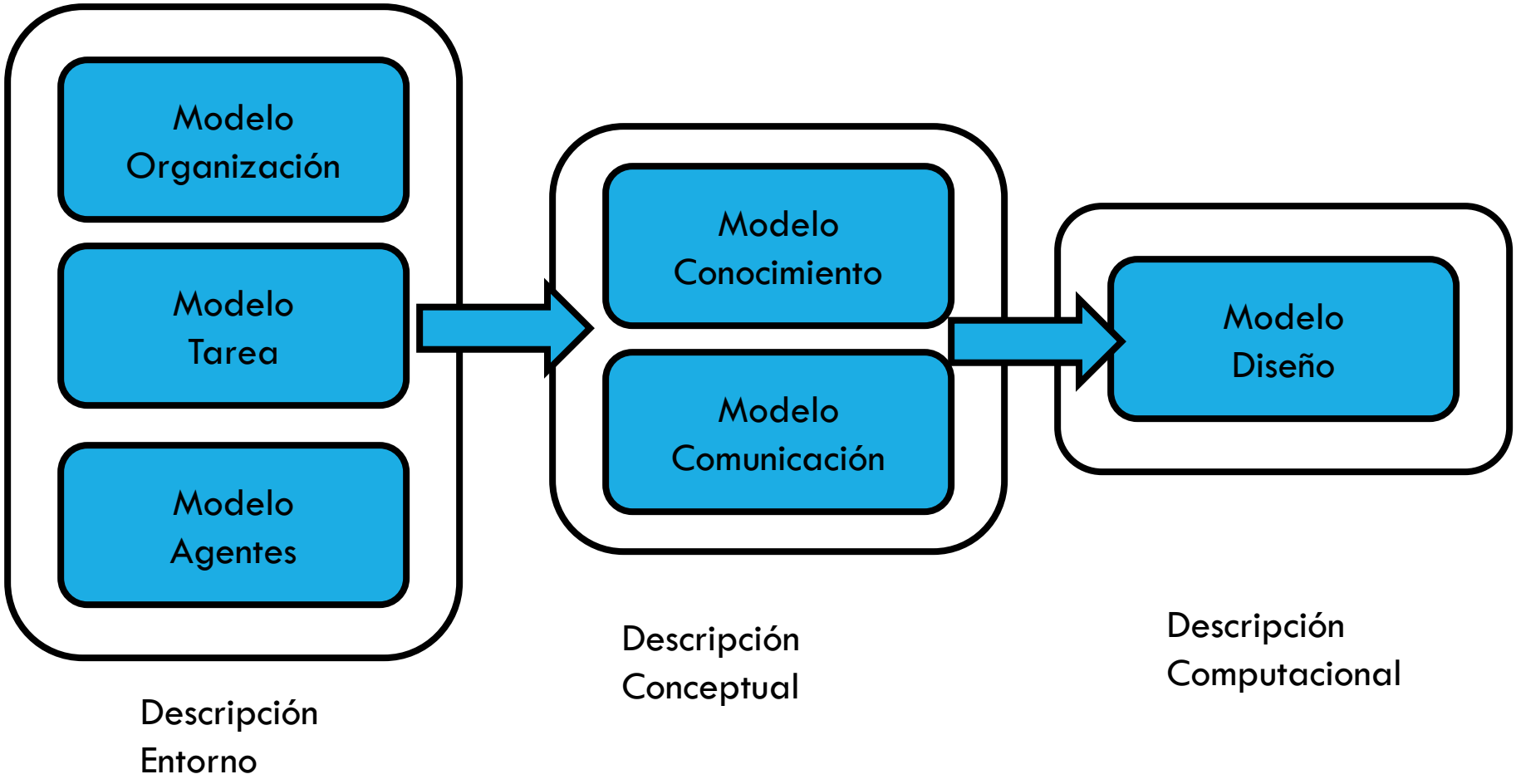
Construir un SBC a partir de diferentes puntos de vista del problema (modelos). Divide y vencerás.

KADS proporciona un conjunto de plantillas-tipo de modelos. Que deben ser configuradas, rellenadas y refinadas durante el proyecto.

El número y el grado de detalle de los modelos depende de la ambición del proyecto y los riesgos.

Evolucionan y se desarrollan en paralelo.

MODELOS DE KADS



ACTORES PARTICIPANTES

Los expertos en el dominio

Los usuarios finales del sistema

El ingeniero de conocimiento

El equipo de desarrolladores

El gestor del proyecto

VISIONES DEL PROBLEMA

Expertos: explica cómo actúa, observación de comportamiento. Escepticismo y poca colaboración.

Usuarios: Proporciona información, recibe explicación. Desconfianza y poca aceptación.

I.C.: Diseña modelos, interacciona con experto y desarrolladores. Sobrevaloración, ser propio experto.

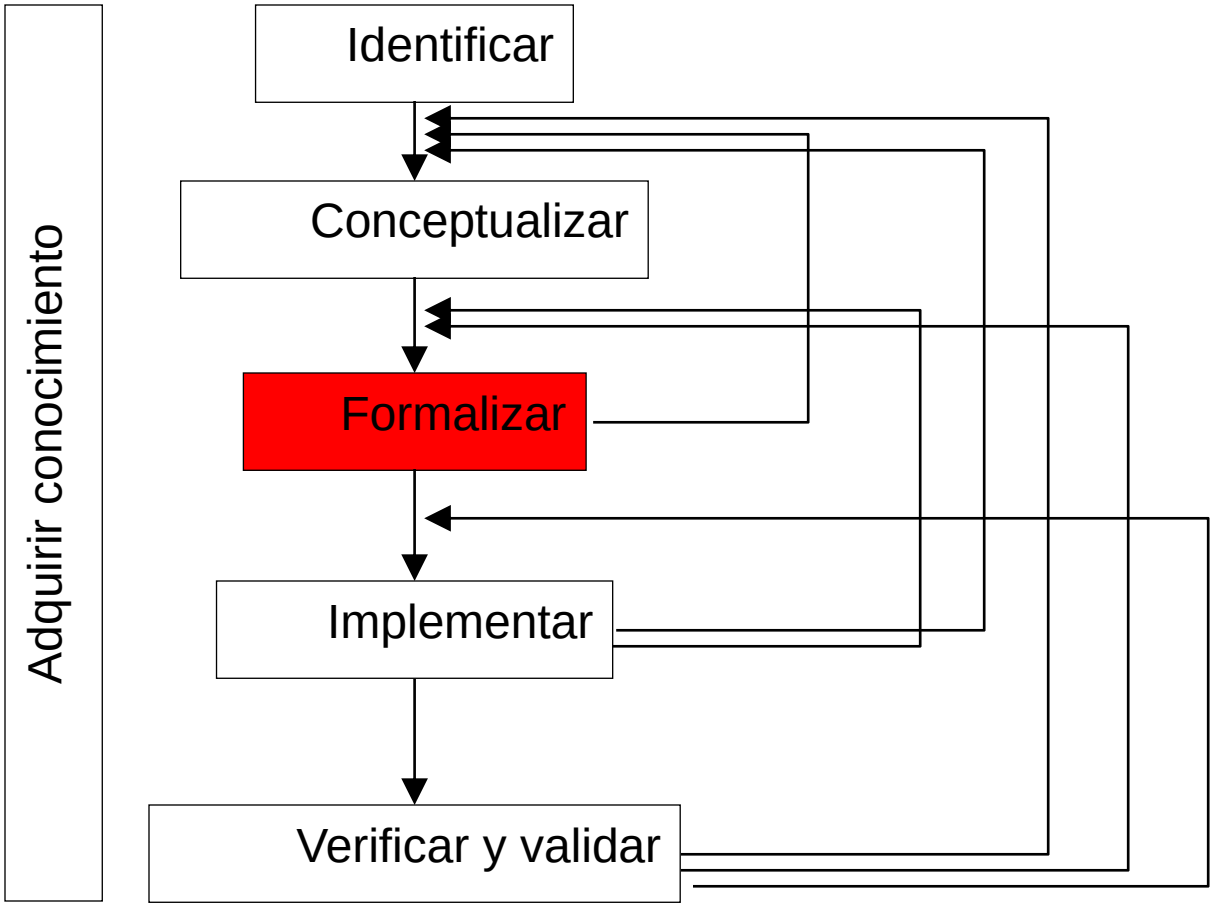
Desarrolladores: Concreta diseño. Problemas implementación.

NIVELES DE CONOCIMIENTO

Alan Newell (1982), diferenció 2 niveles: Simbólico (lenguaje de implementación) y de conocimiento (caracterización del conocimiento y de su uso)

Consecuencia: el conocimiento se debe modelar con independencia de la implementación: Fase de conceptualización entre adquisición e implementación.

METODOLOGÍA DE SS.BB.CC.



ONTOLOGÍAS

Definición
Metodologías
Entornos

ONTOLOGÍAS

¿Qué es una ontología?

- Especificación explícita y formal de una conceptualización compartida (Gruber, 93) (Studer, 98)
- Define los términos y las relaciones básicas para la comprensión de un área, así como las reglas para combinar los términos para definir las extensiones del vocabulario
  - Vocabulario / términos
  - Restricciones/relaciones entre ellos

Proporcionan una forma de representar y compartir el conocimiento utilizando un vocabulario común

- Permiten usar un formato de intercambio de conocimiento
- Proporcionar un protocolo específico de comunicación
- Facilitan la reutilización del conocimiento

COMPONENTES DE UNA ONTOLOGÍA: CONCEPTOS

¿Qué es un Concepto?

- Cualquier cosa sobre lo que se dice algo: objetos físicos o conceptuales, descripción de tareas, funciones, acciones, estrategias, procesos de razonamiento clases en sentido amplio.
- Abstractas o concretas, elementales o compuestas, reales o ficticias...

¿Cómo describir un concepto?

- ¿Metaclases?, ¿particiones?
- Atributos: locales, de instancia, de clase, polimórficos...,
  - Facets: valor por defecto, tipado, restricciones de cardinalidad, documentación asociada, definición operacional, ¿pueden crearse nuevas facets?
  - Jerarquías de atributos

RELACIONES

Relaciones: representan un tipo de interacción entre conceptos del domino.

- Formalmente se definen como subconjuntos del producto cartesiano AxAxA..
  - Inversa, transitividad
  - Objeto, valor.

Composición/Agregación.

Taxonomías: Subclass of

- “disjuntividad”
- “exhaustividad”

Funciones:

- relaciones en intenso, varios a uno.
- ¿arbitrarias o predefinidas?
- ¿argumentos restringidos? ¿se deberían chequear?
- definición operacional

AXIOMAS, REGLAS DE PRODUCCIÓN

Axiomas

- Sentencias del modelo que son siempre ciertas
- Restringen, verifican y deducen nueva información
- Axiomas sobre lógica de primer orden
- Axiomas sobre lógica de segundo orden
- Axiomas como relaciones o restricciones

Reglas de producción

- Conectivas de premisas: And, Or, Not
- Mecanismos de encadenamiento: backward, forward, etc
- Valores de certeza o veracidad
- Procedimientos en antecedente y en consecuente

INSTANCIAS, INDIVIDUOS, HECHOS

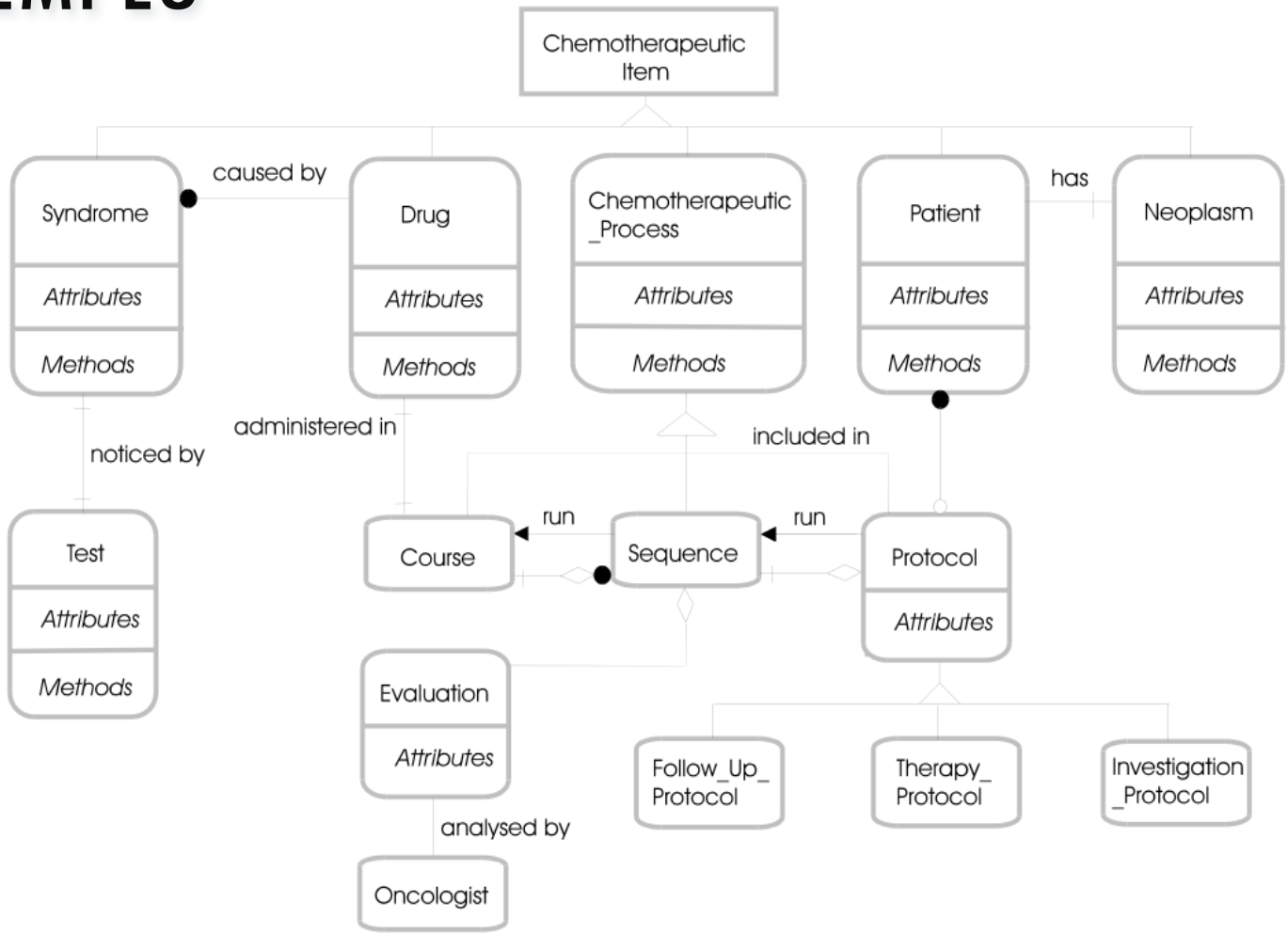
Representan elementos en el dominio

Instancias: Instancias de un concepto

Individuos: Elementos en el dominio que no son clases

Hechos: Relación entre elementos o instancias de relaciones

EJEMPLO


Chemotherapeutic Item
Syndrome
Attributes
Methods
caused by
Drug
Attributes
Methods
Chemotherapeutic _Process
Attributes
Methods
Patient
Attributes
Methods
has
Neoplasm
Attributes
Methods
noticed by
administered in
included in
Test
Attributes
Methods
Course
run
Sequence
run
Protocol
Attributes
Evaluation
Attributes
analysed by
Oncologist
Follow_Up_ Protocol
Therapy_ Protocol
Investigation _Protocol

TECNOLOGÍA ASOCIADA

La tecnología asociada al concepto de ontología supone el desarrollado de herramientas de diferentes tipos [Fensel et al., 2003]:

Lenguajes formales para expresar y representar ontologías

Editores para la construcción semiautomática de nuevas ontologías

Entornos que ayuden a crear nuevas ontologías reutilizando otras previas ya existentes

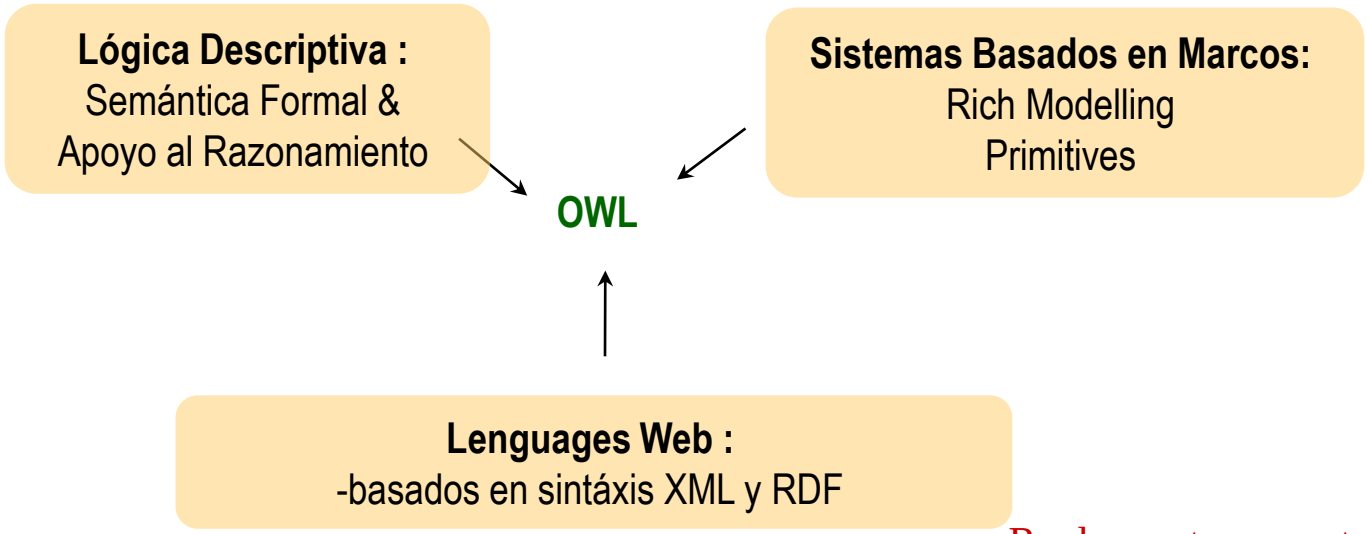
Aprendizaje automático

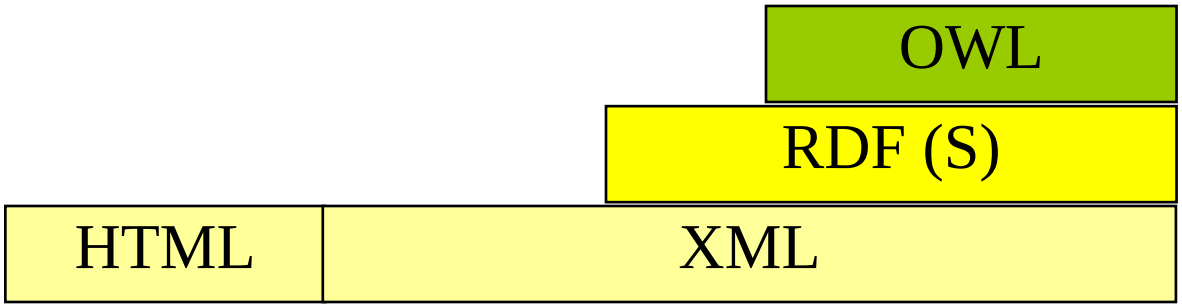
Servicios de inferencia o de razonamiento sobre las ontologías

Herramientas que posibiliten a humanos el acceso inteligente (o navegación inteligente) a la información.

Servicios de traducción o integración entre diferentes ontologías que permiten intercambio multiestándar de datos y el entendimiento entre diferentes definiciones de los mismos conceptos

OWL: ONTOLOGY WEB LANGUAGE





- Reglas y otras construcciones
- Semántica formal y apoyo al razonamiento
- Modelo de datos simple y definición de vocabulario (S)
- Sintaxis

Introducción
Representación
Ciclo de ejecución
Aplicaciones prácticas

SISTEMAS DE PRODUCCIÓN

INTRODUCCIÓN A LOS SISTEMAS DE PRODUCCIÓN

Motivación
Definición
Arquitectura

MOTIVACIÓN

En muchas ocasiones, un experto puede expresar la forma en que resuelve un problema por medio de reglas.

Ya conocemos formalismos que permiten capturar las reglas de forma que se pueden procesar automáticamente (PROLOG en Lógica).

La aplicación de una regla a la situación "actual" genera nuevo conocimiento que puede determinar los siguientes "pasos" de la resolución.

El conocimiento sobre un problema varía: avances científicos, prueba y error de los sistemas, etc. Por ello se requieren sistemas extensibles.

DEFINICIÓN

Sistema que aplica razonamiento basado en reglas, llamadas reglas de producción.

El conocimiento del problema (hechos) está separado del conocimiento de la resolución del problema (reglas)

Las reglas son genéricas para un tipo de problema, los hechos determinan un problema concreto.

Los hechos pueden variar (sistema no monotónico).

Existen lenguajes y entornos de programación específicos (OPS5,CLIPS)

ARQUITECTURA

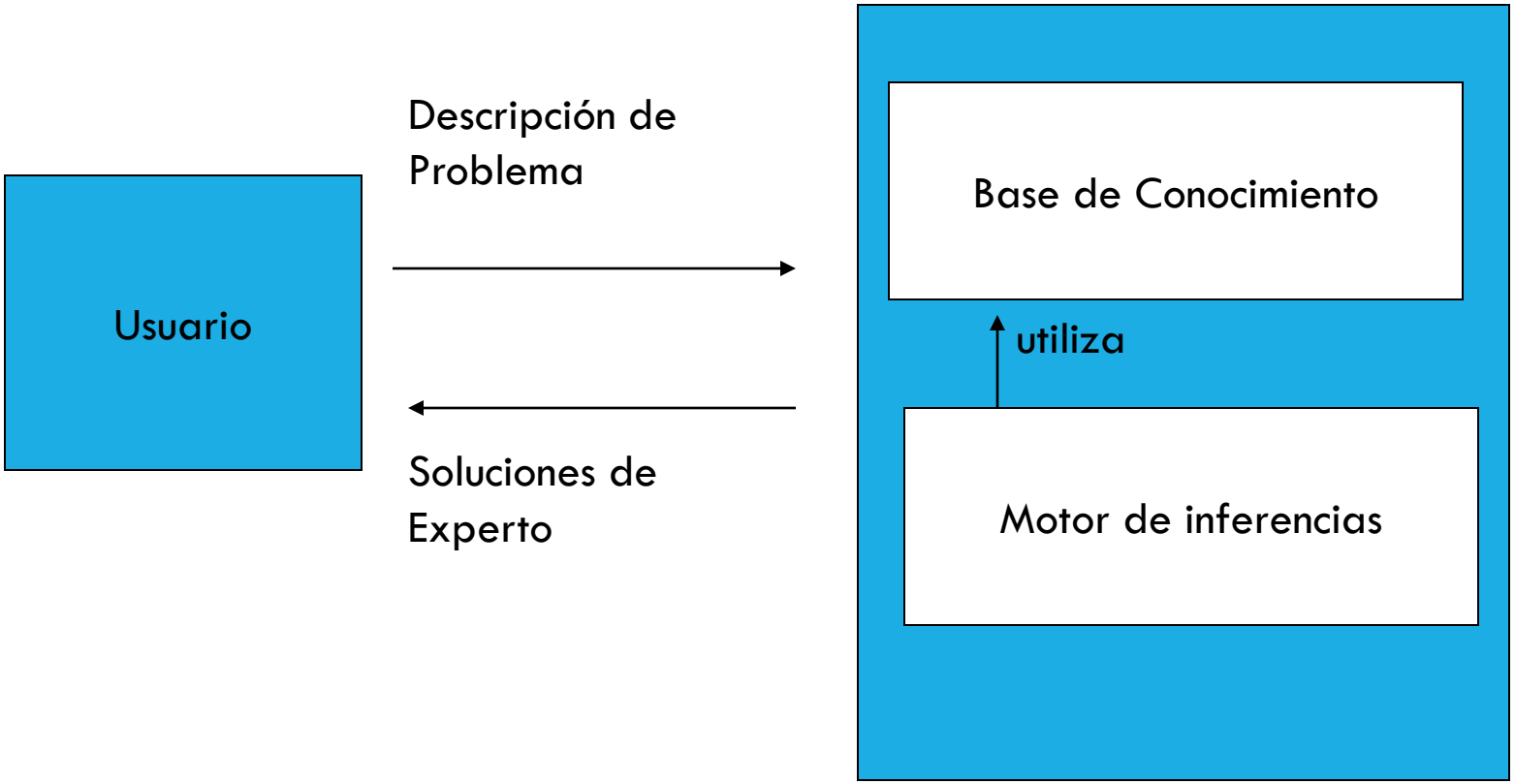
Los elementos del sistema de producción son:

- Base de Hechos (BH): hechos que representan el problema (o el estado de la resolución del mismo).
- Base de Reglas (BR): reglas que representan el conocimiento sobre cómo resolver el problema.
- Motor de inferencia (MI): automatiza el proceso de razonamiento.

Como parte significativa del mismo, hay mecanismos de resolución de conflictos entre reglas.

Los entornos de desarrollo para sistemas de producción incluyen facilidades de interfaz, E/S, funciones de librería, etc.

ARQUITECTURA



REPRESENTACIÓN DEL CONOCIMIENTO EN UN SISTEMA DE PRODUCCIÓN

Cómo se representan los hechos

Cómo se representan las reglas

Ejemplo

HECHOS

Son afirmaciones atómicas, referidas a términos constantes

Pueden verse como una situación de partida; los hechos pueden variar.

Si un hecho no está en la base de hechos, se considera FALSO.

Según el caso, los hechos pueden definirse de forma compleja mediante estructuras (objetos, marcos).

Operaciones: inserción (assert), eliminación o retracto (retract), modificación (modify).

REGLAS

Parte izquierda (LHS): Conjunción de condiciones sobre los hechos de la BH. Las condiciones pueden incluir términos variables.

Parte derecha (RHS): Conjunto de acciones. Puede haber:

- Modificación de los hechos (assert, modify, retract)
- Operaciones de E/S.

Una regla:

IF C1 AND C2 AND….. AND Cm
THEN A1 AND…. AND An

Atención:

- Las reglas no contienen disyunciones ni son estructuras if-then-else.
- Las reglas no contienen condiciones en la parte derecha (RHS), ni disparan directamente otras reglas (pero sí pueden crear hechos que las disparen).

REGLAS

IF C THEN A1 ELSE A2

- IF C THEN A1
- IF NOT C THEN A2

IF C1 OR C2 THEN A

- IF C1 THEN A
- IF C2 THEN A

(Ojo deben ejecutarse las dos)

IF C THEN A1 OR A2

- IF C THEN A1
- IF C THEN A2

(Ojo debe analizarse la precedencia)

EJEMPLO

Una empresa de comercio electrónico se plantea generar automáticamente ofertas de ciertos productos en stock a los buenos clientes, que se determinan según la cantidad de dinero gastada en el portal. Un cliente se considera bueno si ha gastado más de 1000 E.

En el texto se sugieren dos reglas de producción:

- La primera, si hay un cliente que gasta más de 1000 entonces es un buen cliente.
- La segunda, si hay un buen cliente y hay stock de un producto entonces se oferta el producto al cliente.

R1: SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C))

R2: SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)

*Donde P es cualquier producto y C cualquier cliente

EJEMPLO

Supongamos los siguientes hechos en la BH (estado inicial):

en_stock producto1: stock(producto1)

en_stock producto2: stock(producto2)

Marta es un cliente: cliente(marta)

Pedro es un cliente: cliente(pedro)

Juan es un cliente: cliente(juan)

Marta gastó 1200: gastado(marta; 1200)

Pedro gastó 3000: gastado(pedro; 3000)

Juan gastó 500: gastado(juan; 500)

¿Qué sucederá? Dependerá del motor de inferencia

MOTOR DE INFERENCIA

Reducción
Equiparación y activación
Resolución de conflictos
Ejecución

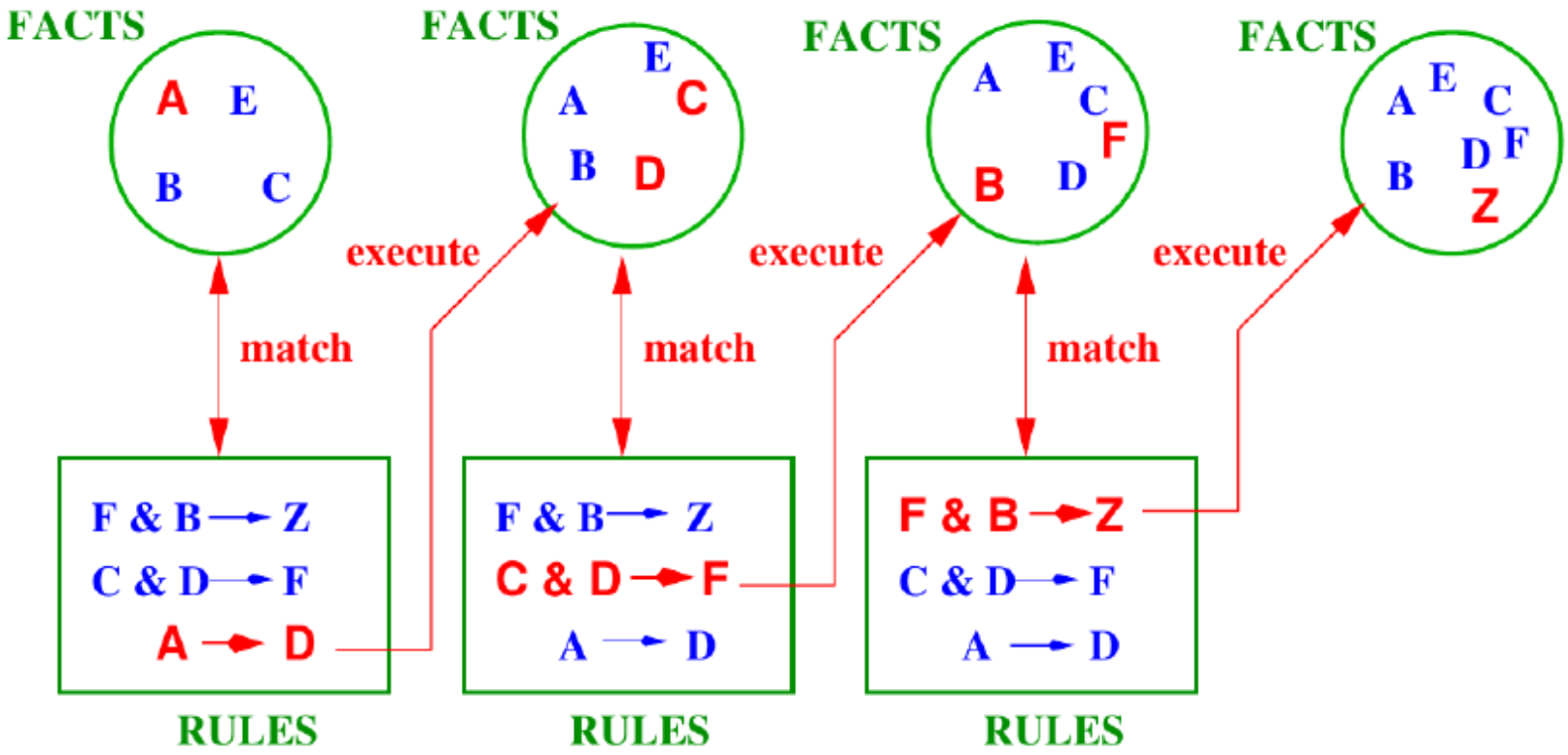
MOTOR DE INFERENCIA. TIPOS

Encadenamiento hacia delante

- Inferencia dirigida por los datos (premisas). Pocos datos iniciales y muchas posibles conclusiones
- Repite el siguiente ciclo: determinar qué reglas tienen su antecedente satisfecho dada la base de hechos actual y ejecutar una de ellas

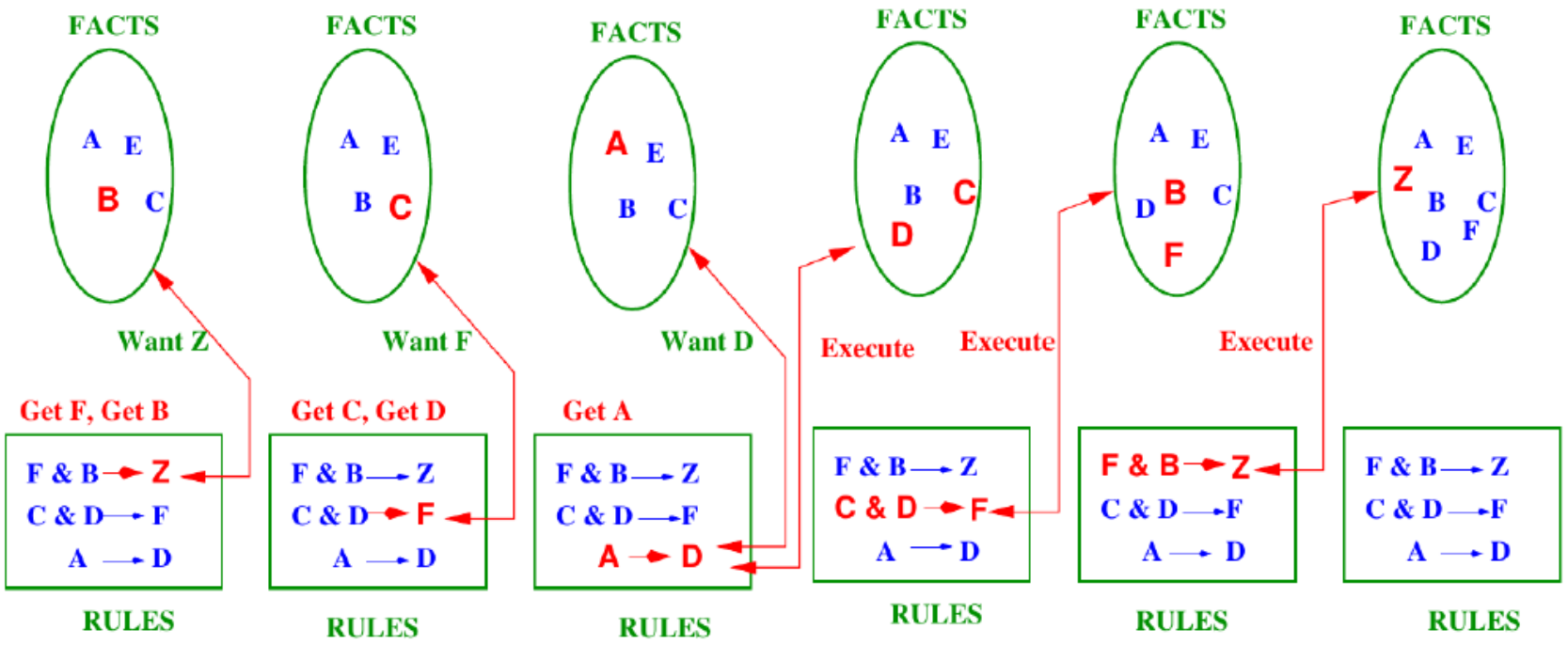
Encadenamiento hacia atrás.

- Inferencia dirigida por las metas (resultados). Muchos datos, pero pocos relevantes
- Se mantiene una lista de (sub)metas por resolver
- Repite el siguiente ciclo: se selecciona una (sub)meta sobre la que trabajar. Si la sub-meta no está en la base de hechos, se buscan las reglas que la generan. Se aplica una de ellas hacia atrás, lo que genera nuevas submetas
- Una vez todas las submetas se han conseguido se ejecutan las reglas hacia delante



MOTOR DE INFERENCIA. TIPOS

Encadenamiento hacia delante



MOTOR DE INFERENCIA. TIPOS

Encadenamiento hacia atrás.

MOTOR DE INFERENCIA. CICLO DE EJECUCIÓN

En un sistema de producción, la ejecución sigue un ciclo definido por las siguientes fases:

- **Reducción** (opcional)
- **Equiparación**. Consiste en determinar qué instancias de las reglas pueden ejecutarse (**activas**).
- **Resolución de conflictos**. Si hay más de una regla activada, escoger qué regla es la que efectivamente se ejecuta.
- **Ejecución**. Ejecutar las acciones especificadas por la regla escogida.

El sistema sólo termina su ejecución cuando la lista de reglas activadas (**agenda**) está vacía o se ejecuta una acción de parar.

EQUIPARACIÓN

El proceso de Equiparación obtiene un conjunto de **instancias** de reglas que pueden ejecutarse.

Una instancia de regla es una regla cuyos términos variables se han reemplazado por constantes.

El proceso de sustitución de los términos variables por constantes se denomina **Instanciación**.

Se marcan como **activas** las instancias de reglas cuya LHS se cumple.

En este proceso una sola regla puede generar muchas instancias activas.

La lista de reglas activas se denomina **conjunto conflicto** o **agenda**.

Consideraciones adicionales:

- **Mundo cerrado**: lo que no está en la BH se considera FALSO. De este modo se pueden tratar condiciones negativas (no hay hechos negados)
- **Refracción**: no se activan instancias que se hayan activado ya: es decir, cada instancia (regla + hechos con los que equipara) se ejecuta una sola vez. Esto evita que el usuario deba prevenir explícitamente los bucles.

EJEMPLO

Base de Hechos

- *stock(producto1)*
- *stock(producto2)*
- *cliente(marta)*
- *cliente(pedro)*
- *cliente(juan)*
- *gastado(marta,1200)*
- *gastado(pedro,3000)*
- *gastado(juan,500)*

Base de Reglas

- R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)
- R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)

EJEMPLO

Equiparación: conjunto conflicto

- R1(C = marta; X = 1200) SI cliente(marta) y gastado(marta,1200) y 1200 > 1000 ENTONCES buen_cliente(marta)
- R1(C = pedro; X = 3000) SI cliente(pedro) y gastado(pedro,3000) y 3000 > 1000 ENTONCES buen_cliente(pedro)

Conjunto Conflicto (CC)

- Se representa mediante la lista de todas las instancias de reglas cuyo antecedente es aplicable dada la base de hechos actual
- Cada instancia se representa con el nombre de la regla y los valores de sus variables
- CC = {R1(C = marta; X = 1200);R1(C = pedro; X = 3000)}

RESOLUCIÓN DE CONFLICTOS

Los sistemas de producción no suelen permitir la ejecución en paralelo de varias reglas.

El proceso de Resolución escoge **una** regla de la agenda para su ejecución (firing)

Si hay más de una regla en la agenda, se suele aplicar una **estrategia de resolución de conflictos** para determinar cuál es la que se ejecuta

Se define una estrategia de ejecución, por ejemplo la primera regla que se activa

Además, el usuario puede definir niveles de prioridad entre reglas; las de mayor prioridad siempre se ejecutan antes.

RESOLUCIÓN DE CONFLICTOS

Estrategias:

- Primera regla
- Profundidad/depth, las nuevas reglas (activadas más recientemente) se ejecutan antes
- Amplitud/breadth, las nuevas reglas se ejecutan después
- Más general/Simplicidad (simplicity), las nuevas reglas se ejecutan antes de las que son igual o más específicas que ellas
- Más específica/Complejidad (complexity), las nuevas reglas se ejecutan antes que las que son igual o menos específicas
- Aleatoria (random), las reglas se ejecutan en orden aleatorio
- Más prioridad
- Más ejecuciones
- Menos ejecuciones
- Explorar todas
- Metarreglas
- Estrategias mixtas

EJECUCIÓN

La primera regla de la agenda se ejecuta

El resultado de la ejecución de una regla puede ser:

- Un conjunto de modificaciones de los hechos
- Un conjunto de modificaciones de las reglas
- Un conjunto de acciones (por ejemplo, E/S)

Puesto que se modifica la base de hechos, tras la ejecución se vuelve a calcular las reglas que se activan.

EJEMPLO - PROFUNDIDAD

Base de Hechos

- *stock(producto1) stock(producto2)*
- *cliente(marta) cliente(pedro)*
- *cliente(juan) gastado(marta, 1200)*
- *gastado(pedro, 3000) gastado(juan, 500)*

Base de Reglas

- *R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)*
- *R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)*

Conjunto conflicto

- *R1(C = marta; X = 1200)*
- *R1(C = pedro; X = 3000)*

EJEMPLO - PROFUNDIDAD

Base de Hechos

- *stock(producto1) stock(producto2)*
- *cliente(marta) cliente(pedro)*
- *cliente(juan) gastado(marta, 1200)*
- *gastado(pedro, 3000) gastado(juan, 500)*

Base de Reglas

- *R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)*
- *R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)*

Conjunto conflicto

R1(C = marta; X = 1200)

- *R1(C = pedro; X = 3000)*

EJEMPLO - PROFUNDIDAD

Base de Hechos

- *stock(producto1) stock(producto2)*
- *cliente(marta) cliente(pedro)*
- *cliente(juan) gastado(marta, 1200)*
- *gastado(pedro, 3000) gastado(juan, 500)*

buen_cliente(marta)

Base de Reglas

- *R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)*
- *R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)*

Conjunto conflicto

- *R1(C = marta; X = 1200)*
- *R1(C = pedro; X = 3000)*

EJEMPLO - PROFUNDIDAD

Base de Hechos

- *stock(producto1) stock(producto2)*
- *cliente(marta) cliente(pedro)*
- *cliente(juan) gastado(marta, 1200)*
- *gastado(pedro, 3000) gastado(juan, 500)*
- *buen_cliente(marta)*

Base de Reglas

- *R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)*
- *R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)*

Conjunto conflicto

- *R1(C = marta; X = 1200)*
- *R1(C = pedro; X = 3000)*

R2(C = marta; P = producto1)

- *R2(C = marta; P = producto2)*

EJEMPLO - PROFUNDIDAD

Base de Hechos

- *stock(producto1) stock(producto2)*
- *cliente(marta) cliente(pedro)*
- *cliente(juan) gastado(marta, 1200)*
- *gastado(pedro, 3000) gastado(juan, 500)*
- *buen_cliente(marta)*

oferta(marta, producto1)

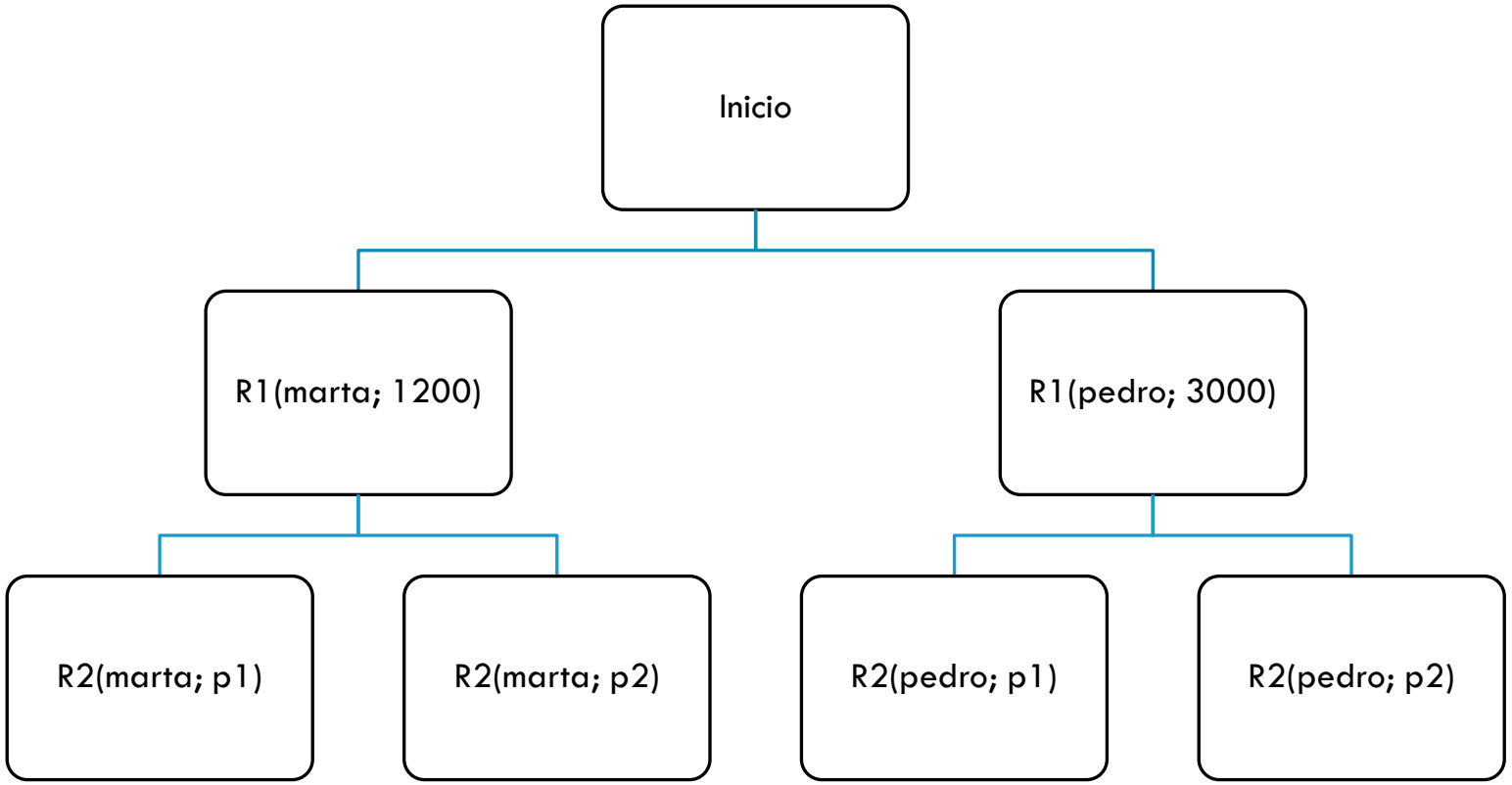
Base de Reglas

- *R1 SI cliente(C) y gastado(C,X) y X > 1000 ENTONCES buen_cliente(C)*
- *R2 SI buen_cliente(C) y stock(P) ENTONCES oferta(C,P)*

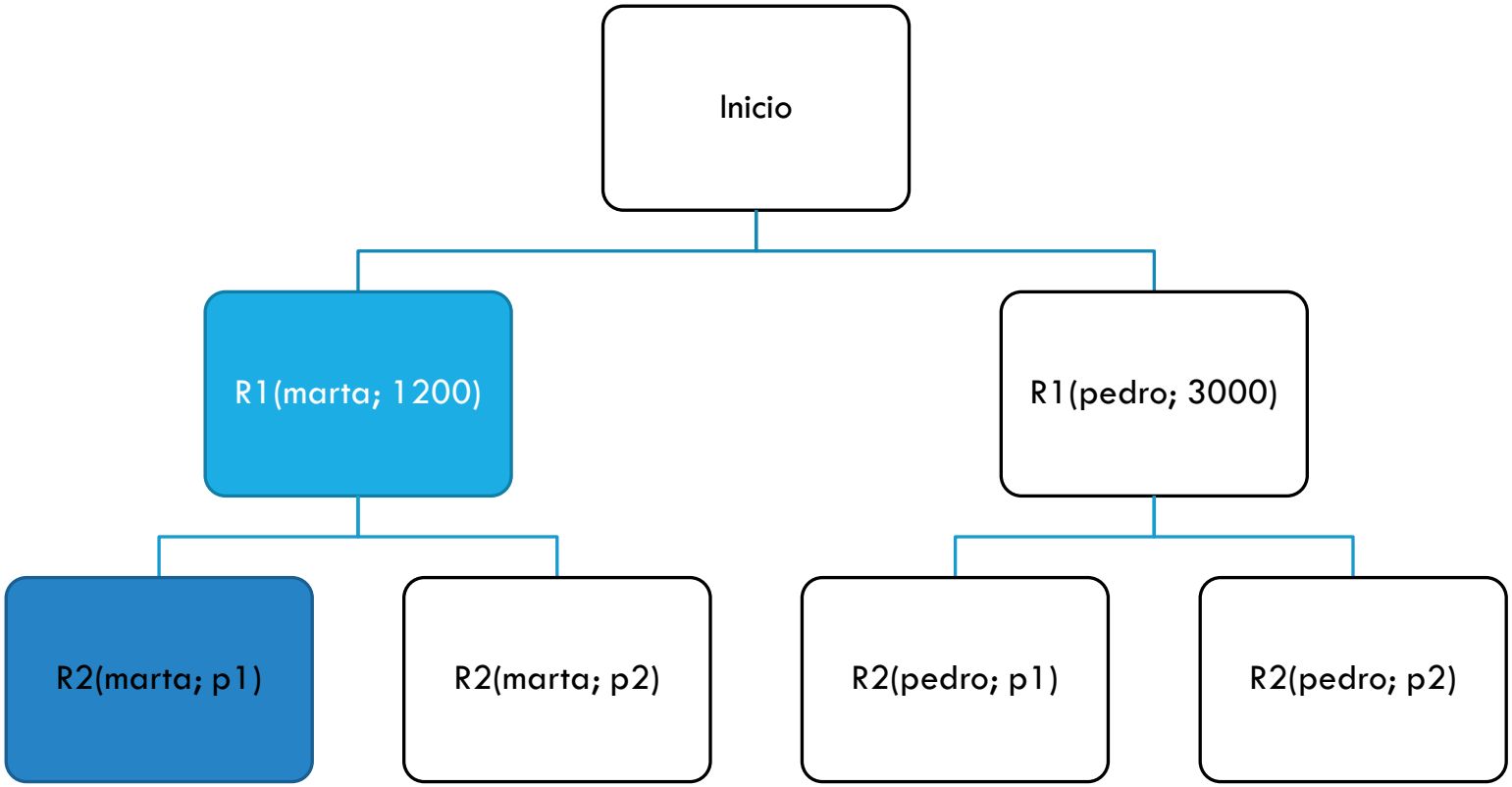
Conjunto conflicto

- *R1(C = marta; X = 1200)*
- *R1(C = pedro; X = 3000)*
- *R2(C = marta; P = producto1)*
- *R2(C = marta; P = producto2)*

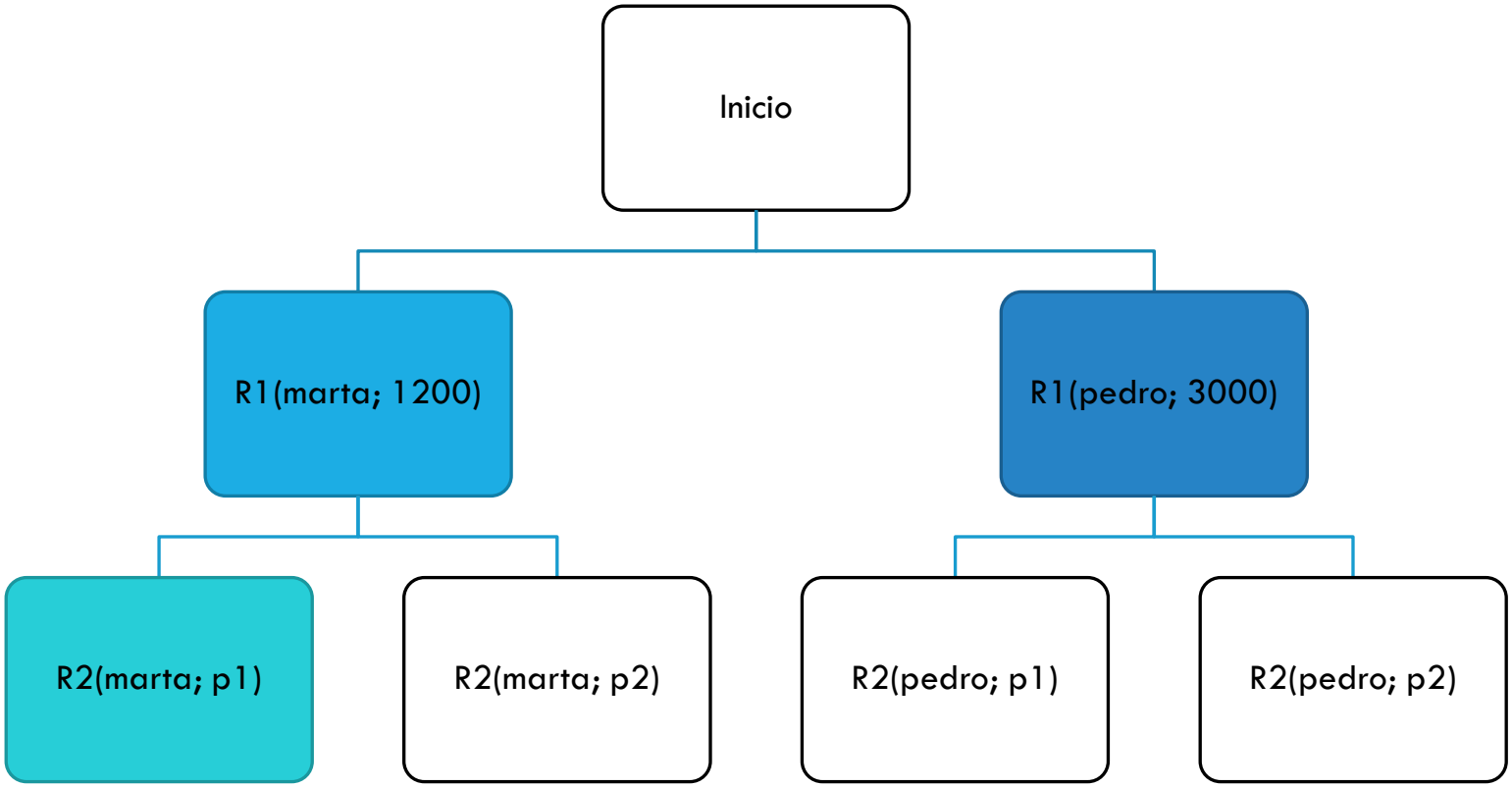
EJEMPLO – EN MODO ÁRBOL



EJEMPLO – PROFUNDIDAD



EJEMPLO – AMPLITUD



EQUIPARACIÓN MÁS EFICIENTE

Aproximación inicial: calcular el conjunto conflicto completamente en cada ciclo

Problema: lento

Solución: algoritmo RETE (redundancia temporal)

Establece un grafo a partir de las reglas (red RETE)

Propaga los hechos iniciales por la red

Cuando los hechos cambian, los cambios se propagan por la red

En cada ciclo, la red proporciona el CC en sus nodos finales

Idea clave: similitud estructural

Tiempo vs. memoria

LOS SISTEMAS DE PRODUCCIÓN EN LOS SISTEMAS REALES

Ventajas e inconvenientes

Ejemplos de aplicaciones

VENTAJAS E INCONVENIENTES

Ventajas:

- Permiten una representación natural del conocimiento.
- Se puede agregar conocimiento de forma modular (añadiendo reglas).
- Programación **declarativa**: no es preciso centrarse en cómo resolver el problema, sino en qué se conoce sobre el mismo.

Inconvenientes

- Posibilidad de ineficiencia, al tener que generar muchas instancias de reglas para luego descartar y repetir el ciclo
- Adaptación a un modelo de programación declarativo.
- Dificultades para programar sin estructuras de control.
- Algunos problemas específicos: Encadenamiento infinito, gestión de hechos contradictorios, modificación de reglas existentes.

APLICACIONES

Asistentes

- Un ejemplo de sistema experto guiado por el usuario (asistentes de Windows, por ejemplo).
- Las respuestas se piden a través del interfaz.
- A partir de la entrada del usuario, se encamina al mismo hacia la información necesaria para la solución.
- Análogo a un sistema de diagnóstico médico: preguntas sobre síntomas, y medidas sobre el paciente.
- Razonamiento mediante encadenamiento hacia delante.

APLICACIONES

Sistemas Expertos

- Sistemas que intentan capturar el conocimiento existente de personas experimentadas en un dominio, y reproducir su forma de razonamiento para obtener un resultado (Durkin, 1994).
- Ejemplos clásicos: MYCIN, Dendral, Caduceus, R1=XCON
- Lenguaje específico: OPS5
- Aplicaciones: concesión de hipotecas, autorización de crédito, detección de fraude, comercio electrónico, diagnóstico, asistencia al usuario, etc.

APLICACIONES

Planificación

- El sistema intenta encontrar un **plan** que conduzca desde el estado inicial hasta el estado deseado.
- Hay múltiples ejemplos académicos: mundo de bloques, mono y plátano, etc.
- El objetivo es llegar a un determinado estado del mundo: mono come el plátano, o disposición de los bloques.
- Sistema experto guiado por objetivos.
- Para alcanzar el objetivo hay que cumplir unos objetivos previos; las reglas saben cuáles son (para abrir un cofre, hay que tener la llave; para mover objeto, hay que cogerlo primero)
- Las reglas no especifican cada caso, sólo el comportamiento general que permite completar un subobjetivo sea cual sea el caso concreto.
- Razonamiento mediante (o simulando) encadenamiento hacia atrás