

TEMA 2: CommonKADS Modelos Conceptuales



1. El modelo del conocimiento.
 - a. Construcción de los modelos de conocimiento
 - b. Plantillas de modelos de conocimiento. Elementos reutilizables
 - c. Conocimiento de la tarea
 - d. Conocimiento inferencial
 - e. Conocimiento del dominio
2. El modelo de comunicación
3. Un caso de estudio

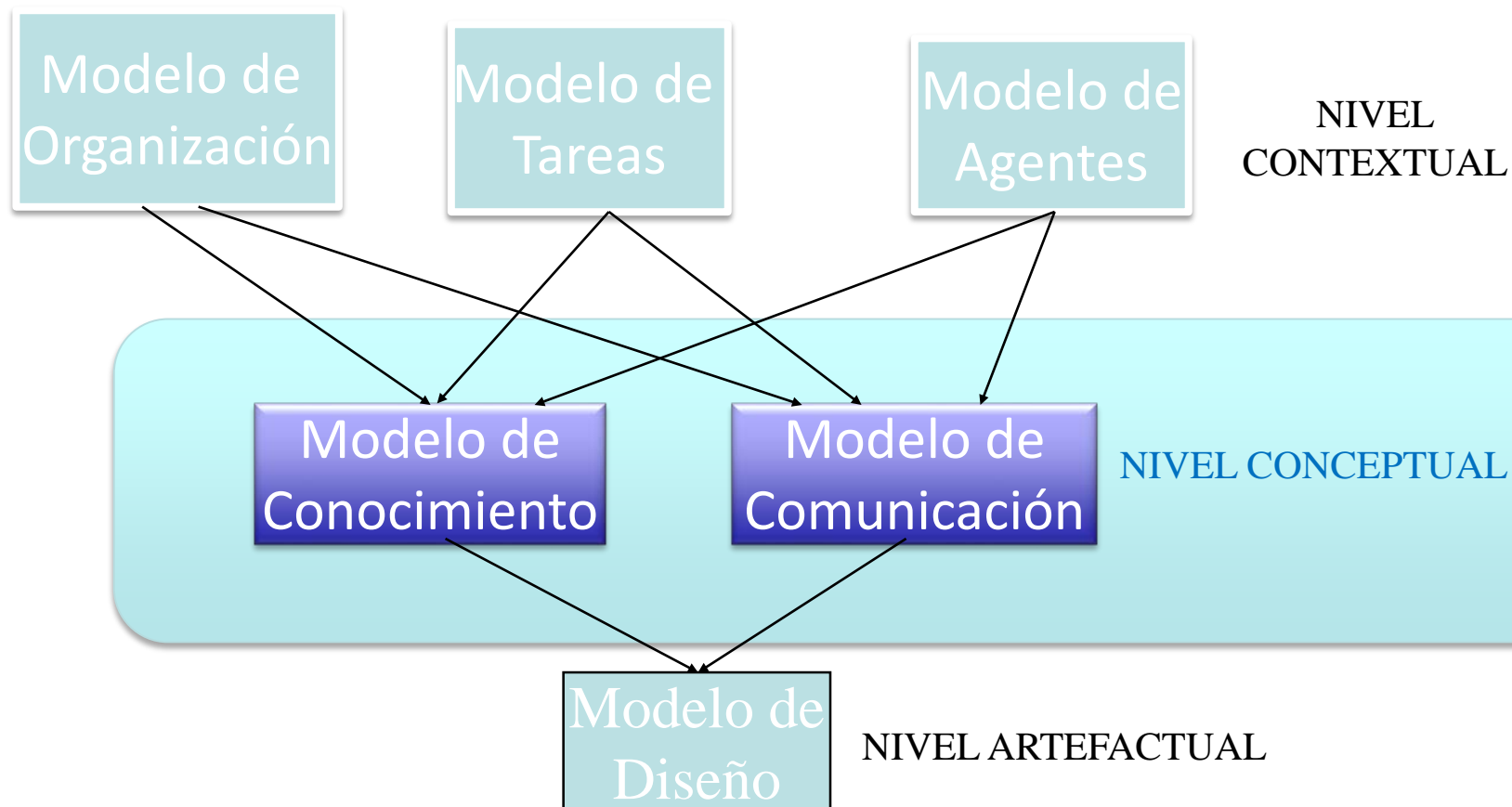


- Entender la necesidad y el interés del modelado conceptual
- Entender la visión de IC como actividad de modelado y la adquisición de conocimiento como un proceso de construcción de modelos.
- Entender los constructos básicos del modelado del conocimiento.
- Entender el modelo de comunicación
- Aplicar los conceptos al modelado de conocimiento de una tarea concreta.
- Aplicación directa a la parte práctica de la asignatura

- G.Schreiber et col. “Knowledge engineering and management”. MIT 2000.
- A. Alonso Betanzos et col. “Ingeniería de Conocimiento. Aspectos Metodológicos”. Pearson Educación, 2004.



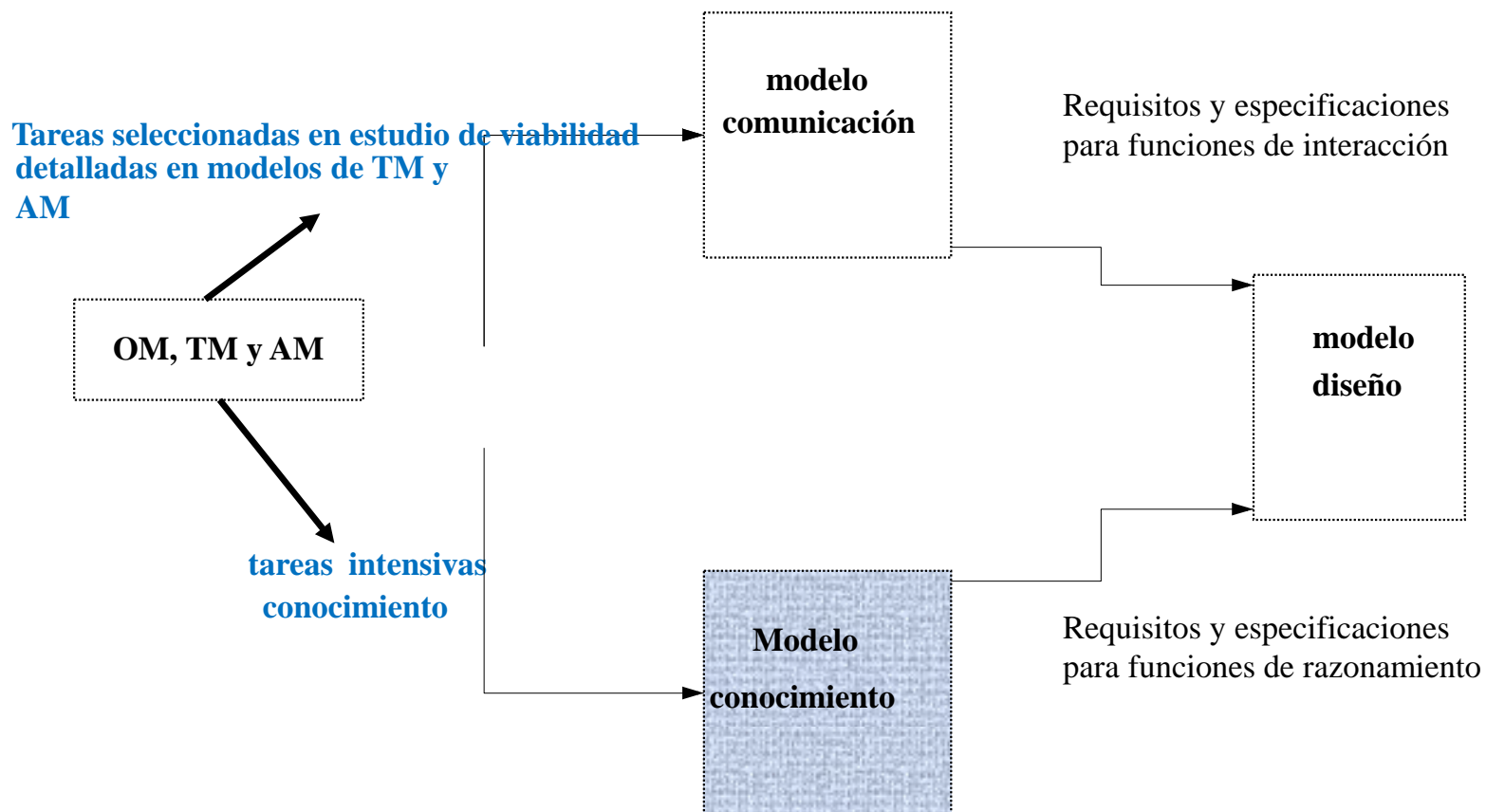
EL MODELO DEL CONOCIMIENTO.



- Es un proceso cíclico
- Es una actividad sin fin
- Depende de la visión subjetiva del ingeniero de conocimiento
- Necesita de una fase de evaluación

Modelo de conocimiento

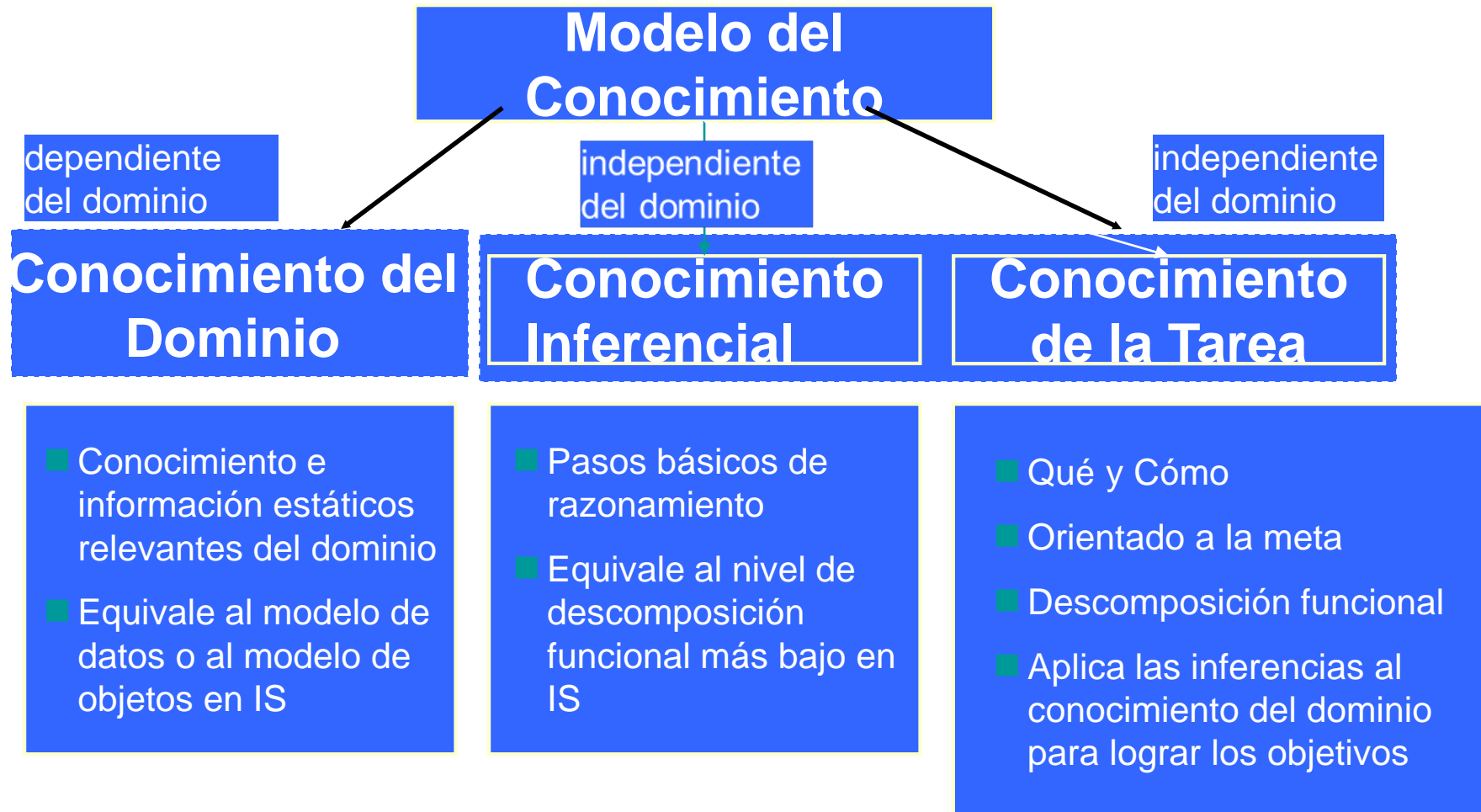
- Es una herramienta especializada para **especificar tareas** en **dominios intensivos** en conocimiento.
- Especifica los **requisitos de conocimiento y razonamiento** del sistema futuro.
- **No** se incluyen aspectos de **comunicación** del sistema
- **No** contiene términos específicos de la **implementación**
- Su estructura es similar a la de los modelos de análisis tradicional en IS.
- Son un aspecto importante para la **reutilización** del software

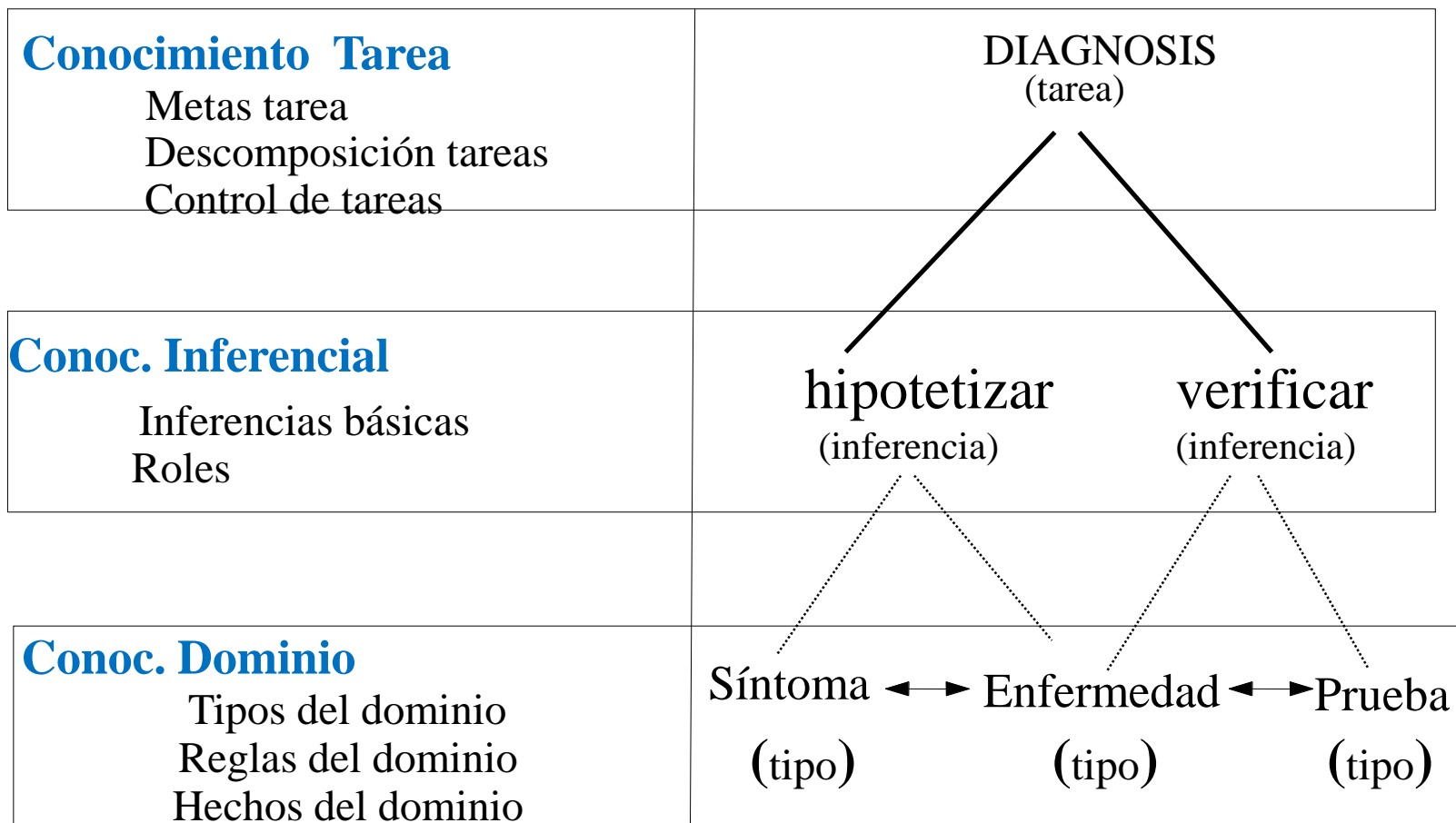


Modelo de Organización	Formulario OM-1: Problemas y Posibilidades de Mejor
Problemas y Oportunidades	Uno de los problemas que existen en el IES es la necesidad de planificar, al inicio del curso los horarios completos (los de profesores, aulas, materias, guardias y tutorías). Esta planificación es compleja, debido a que necesitan cumplirse ciertos requisitos y restricciones (disponibilidad de los profesores, capacidad de aulas, limitación.....)
Contexto Organizacional
Soluciones	La solución que se propone es desarrollar un SBC que genere automáticamente y de forma completa y correcta todos los horarios necesarios del IES a partir de unos requisitos iniciales de los recursos y de las preferencias proporcionadas por el equipo docente.

Modelado contextual. Ejemplo horarios OM-3

Modelo de organización		Formulario OM-3: Descomposición de los Procesos				
<i>nº</i>	<i>Tarea</i>	<i>Realizada por</i>	<i>¿Dónde?</i>	<i>Recursos de conocimiento</i>	<i>¿Intensiva en conocimiento?</i>	<i>Importancia</i>
1	Determinar los recursos del IES	Equipo directivo	IES a partir de las bases de datos disponibles	No	No	Requisito necesario para iniciar el proceso
2	Descripción de los requisitos obligatorios	Equipo directivo	IES	Reglamento de la Xunta de Galicia	No	Requisito necesario
3	Reunir las preferencias del profesorado	Equipo directivo	IES	No	No	Opcional
4	Generación de horarios conforme a los requisitos	Director y equipo directivo	IES	Experiencia en la generación de horarios	Sí (elevado)	Paso clave
5	Revisión/ Mejora del horario	Equipo directivo	IES	Experiencia en la generación de horarios	Moderado	Recomendable

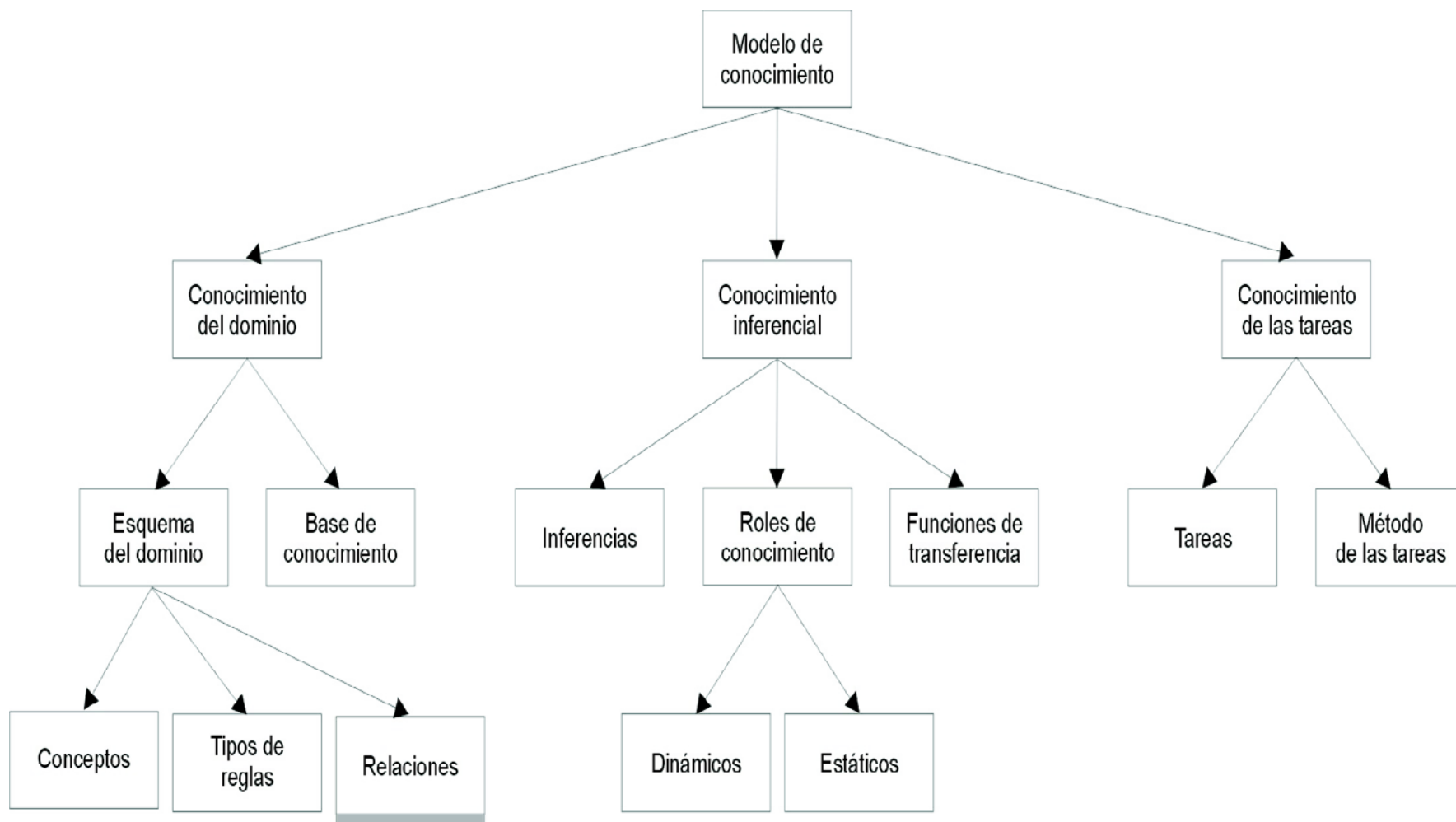




CONSTRUCCIÓN DEL MODELO DE CONOCIMIENTO

- El modelo de conocimiento es un producto del modelado.
- Complejo:
 - ¿Cómo llevar a cabo el proceso de construcción del modelo?
 - Cuello botella para personal con escasa experiencia
- Solución: Modelo del proceso
 - tan prescriptivo como sea posible
 - elementos del proceso: estado, actividad, guías, técnica
- El modelado es una actividad constructiva
 - no existe una solución correcta única ni un camino óptimo
- El modelado de conocimiento es una forma especializada de especificación de requisitos
 - se usan los principios generales de la IS.

Modelo de conocimiento. Subpartes del mismo



ESTADOS

ACTIVIDADES TÍPICAS

Identificación conocimiento

- Familiarización con el dominio
(fuentes de información, glosario, escenarios típicos)
- Lista potencial de los componentes del modelo reutilizables
(componentes relacionados con la tarea y con el dominio)

Especificación conocimiento

- Escoger plantilla de tarea
(proporciona la descomposición inicial de la tarea)
- Construir la conceptualización inicial del dominio
(principales tipos de información del dominio)
- Especificación completa del modelo del dominio
(modelo del conocimiento con BC parciales)

Refinado conocimiento

- Validar modelo conocimiento
(simulación en papel, prototipo de razonamiento del sistema)
- Refinado de la base de conocimiento
(completar las bases de conocimiento)

- **Meta**

- estudiar los items de conocimiento
- prepararlos para su especificación

- **Entrada**

- tarea intensiva en conocimiento seleccionada
- principales items de conocimiento identificados
- se parte de los elementos en TM-2
- Identificación de la tarea de la aplicación
asesoramiento, configuración, combinación de tipos de tareas,

- **Actividades**

- explorar y estructurar las fuentes de información
- estudiar la naturaleza de la tarea en más detalle

TM-2 del ejemplo de horarios

Modelo de Tareas	Formulario TM-2: Elemento de Conocimiento	
Nombre	Experiencia en la generación de horarios	
Poseído por	Director	
Usado en	Tarea 4 de OM-3	
Dominio	Experiencia	
Naturaleza del conocimiento	(Sí/No)	¿Cuello de botella/debe ser mejorado?
Formal, riguroso	No	No
Empírico, cuantitativo	No	No
Heurístico, sentido común	Sí	Sí
Altamente especializado, específico del dominio	Sí	Sí
Basado en la experiencia	Sí	Sí
Basado en la acción	No	No
Incompleto	Sí	No
Incierto, puede ser incorrecto	Sí	No
Cambia con rapidez	No	No
Difícil de verificar	No	NO
Tácito, difícil de transferir	Sí	Sí
Forma del conocimiento		
Memorial	Sí	Sí
Papel	No	No
Electrónica	No	No
Habilidades	No	No
Otros	No	No
Disponibilidad del Conocimiento		
Limitaciones en tiempo	Sí	Sí
Limitaciones en espacio	No	No
Limitaciones en acceso	Sí	Sí
Limitaciones en calidad	Sí	Sí
Limitaciones en forma	No	No

Fase 1.- Identificación del conocimiento

- Actividad 1.1.- Búsqueda de las fuentes de información.
- Actividad 1.2.- Listado de los componentes reutilizables.

Actividad 1.1.- Búsqueda de las fuentes de información.

■ Factores

- Naturaleza de las fuentes

¿están claramente entendidas?, ¿tienen base teórica?

- Diversidad de las fuentes

no existe una fuente de información única (ej. libro o manual)

diversas fuentes pueden ser conflictivas

los expertos múltiples son un factor de riesgo.

■ Técnicas

- marcar textos en fuentes de información clave
- realizar entrevistas estructuradas para clarificar los posibles “agujeros”

■ Problema principal:

- balance entre aprender sobre el dominio y convertirse en un experto en él.

- Entrevistas con el personal de la organización que tiene que hablar con los expertos, pero que no son expertos.
- Evitar sumergirse en teorías complicadas y detalladas, a no ser que esté probada su utilidad.
- Construir unos cuantos escenarios típicos que se entiendan a nivel global.
- No pasar demasiado tiempo en esta actividad.
 - Dos semanas/persona debe ser el máximo.

■ Tangible

- Listado de fuentes de conocimiento del dominio, incluyendo una pequeña caracterización
- Resúmenes de textos clave seleccionados.
- Glosario de términos
- Descripción de los escenarios desarrollados

■ Intangible

- El propio entendimiento y comprensión del dominio por parte del Ingeniero de Conocimiento
 - resultado muy importante

- **Búsqueda de las fuentes de información**
 - Repasar formularios OM-3, OM-4 y TM-2
 - Las tareas en las que se va a centrar el desarrollo del modelo de conocimiento son
 - Generación de horarios conforme a los requisitos
 - Revisión/mejora del horario
 - Ambas usan el mismo recurso de conocimiento
 - Experiencia en la generación de horarios
 - Entrevistamos de nuevo al director del IES para preguntarle cómo se realiza esta tarea.

■ Glosario:

- **Bloque:** Conjunto de horas que engloba una mañana o una tarde. Por ejemplo, el Lunes por la mañana es un bloque y el Lunes por la tarde otro.
- **Ciclo:** Engloba a todos los grupos de un mismo nivel. Por ejemplo, todos los 1º de ESO son un ciclo.
- **Condición obligatoria:** Norma que los horarios deben cumplir de forma obligatoria. Por ejemplo, “Ningún profesor puede tener un día” entero libre.
- **Condición recomendable:** Norma que los horarios deben respetar en la medida de lo posible. Por ejemplo, “Es preferible que ningún profesor tenga más de seis horas al día”.
- **Desdoble de un grupo:** Situación en la que alumnos de un mismo grupo han elegido materias optativas distintas de manera que, a ciertas horas hay que dividirlos en varios grupos con aulas y profesores distintos.
- **Desdoble de una materia:** Es una materia en la que en alguna hora divide a sus alumnos en varios grupos con profesores distintos y en aulas separadas. Por ejemplo, en inglés, de las 3 horas semanales, 1 hora es para inglés oral, y se deben hacer dos grupos de 15 alumnos.
- **Desglose:** Es la división por días de las horas semanales de una asignatura. Por ejemplo, Inglés con (4 horas semanales) tiene un desglose 1 2 1, es decir, hay un día que tiene 2 horas seguidas.
- **Enseñanza:** Es el tipo de titulación. En el centro en estudio se imparte primer ciclo de ESO, segundo ciclo de ESO, Bachillerato, Garantía social, Ciclos de diurno y Ciclos de nocturno.
- **Grupo:** Es un conjunto de alumnos con un horario, profesores y aulas asignadas. Por ejemplo, 1ºA ESO.
- **Guardia:** Es una hora en la que el profesor no imparte clase, pero tiene que estar en el centro realizando tareas de vigilancia.
- **Horario de aulas (HA):** Horario con todas las aulas del IES que indica las horas a las que está ocupada cada aula y por quién.
- **Horario de grupos (HG):** Horario para cada grupo que para cada hora especifica qué materia se imparte, por qué profesor y en qué aula.
- **Horario de guardias (HGu):** Horario para observar qué profesores están de guardia a cada hora del horario general de guardias.
- **Horario de profesores (HP):** Horario que describe para cada profesor a qué hora tiene clase, de qué materia, con qué grupo y en qué aula.

Glosario

- **Horario general de guardias (HGG):** Horario que comprende todas las horas en las que tiene que haber algún profesor de guardia.
- **Horario general del centro (HGC):** Días y horas en las que está abierto el centro. Las distintas enseñanzas impartidas en el centro se tienen que adaptar al HGC.
- **Materia:** Conjunto de horas que imparte un determinado profesor a un determinado grupo. Por ejemplo, Matemáticas en 1ºA ESO por Ana González.
- **Preferencia de profesor:** Conjunto de horas del horario que un profesor prefiere no tener clase. Por ejemplo, José González Lesmes, prefiere no entrar los Lunes a primera hora de la mañana.

Escenario

Se trata de un IES en el que el horario es de 10:00 a 13:00 de lunes a viernes dividido en franjas de una hora. Hay dos grupos (1ºA y 1ºB). Los dos grupos tienen 3 materias (Física, Matemáticas y Lingua galega). Hay un profesor por cada materia, es decir, dos grupos no pueden tener la misma materia a la misma hora. Cada materia tiene 5 horas semanales de docencia. Ningún profesor puede tener un día libre. Es preferible que ningún profesor tenga toda la mañana con el mismo grupo. El profesor de matemáticas no quiere impartir la hora del lunes a las 10:00 y el de física prefiere no entrar ningún día a las 10:00.

Fase 1.- Identificación del conocimiento

- Actividad 1.1.- Búsqueda de las fuentes de información.
- Actividad 1.2.- Listado de los componentes reutilizables.

Actividad 1.2: Lista potencial de componentes reutilizables

- Meta: allanar el camino para componentes reutilizables.
- Dos aspectos en reutilización:
 - Dimensión tarea
 - elegir Tarea tipo asignada en el TM
 - construir una lista de plantillas de tareas
 - Dimensión dominio
 - tipo de dominio: ej. dominio técnico
 - buscar descripciones estándar
 - librerías de ontologías, librerías de modelos de referencias, librerías de modelos de productos, etc.

■ Listado de los componentes reutilizables

- Dimensión tarea:
 - Recurriremos a las plantillas de la librería de tareas de CommonKADS.
- Dimensión dominio:
 - Se podrán reutilizar además en posteriores usos todos los datos introducidos:
 - Recursos del IES (profesores, aulas, etc.)
 - Requisitos (obligatorios y preferibles)
 - Preferencias de los profesores
 - Relaciones entre materias y grupos
 - Todos los tipos de condiciones entre materias (p.ej. que coincidan en la misma hora)
 - Todo esto se almacena en una base de datos relacional

ESTADOS

ACTIVIDADES TÍPICAS

Identificación conocimiento

- Familiarización con el dominio
(fuentes de información, glosario, escenarios típicos)
- Lista potencial de los componentes del modelo reutilizables
(componentes relacionados con la tarea y con el dominio)

Especificación conocimiento

- Escoger plantilla de tarea
(proporciona la descomposición inicial de la tarea)
- Construir la conceptualización inicial del dominio
(principales tipos de información del dominio)
- Especificación completa del modelo del dominio
(modelo del conocimiento con BC parciales)

Refinado conocimiento

- Validar modelo conocimiento
(simulación en papel, prototipo de razonamiento del sistema)
- Refinado de la base de conocimiento
(completar las bases de conocimiento)

■ Meta:

- Completar la especificación de conocimiento
 - excepto para los contenidos de los modelos del dominio.
 - Los modelos del dominio necesitan sólo contener instancias ejemplo.

■ Actividades

- Elegir una plantilla de tarea.
- Construir una conceptualización inicial del dominio.
- Especificar las tres categorías del conocimiento.

- Actividad 2.1.- Selección de la plantilla de tareas.
- Actividad 2.2.- Especificación inicial del esquema de conocimiento del dominio.
- Actividad 2.3.- Completar la especificación del modelo de conocimiento.

- Reutilización
 - Eficacia, calidad asegurada.

- Criterio de selección: Características de la tarea de la aplicación.
 - Naturaleza de la salida: categoría por defecto, plan
 - Naturaleza de las entradas: tipos de datos disponibles
 - Naturaleza del sistema
 - Restricciones del contexto de la tarea:
requisitos certidumbre, costes de las observaciones.

- Guías para la selección
 - Construir estructura inferencial anotada (y esquema básico del dominio)
 - Una plantilla incompleta es mejor que no tener plantilla

- Las plantillas de las tareas son una combinación reutilizable de elementos del modelo:
 - Tareas,
 - Inferencias
 - Dominio
- Rango de tipos de tareas limitado en IC
 - Se usa una jerarquía basada en el término “sistema”
- Las plantillas de las tareas son una especie de patrones de diseño para tareas intensivas en conocimiento.

Plantillas de conocimiento. Elementos reutilizables

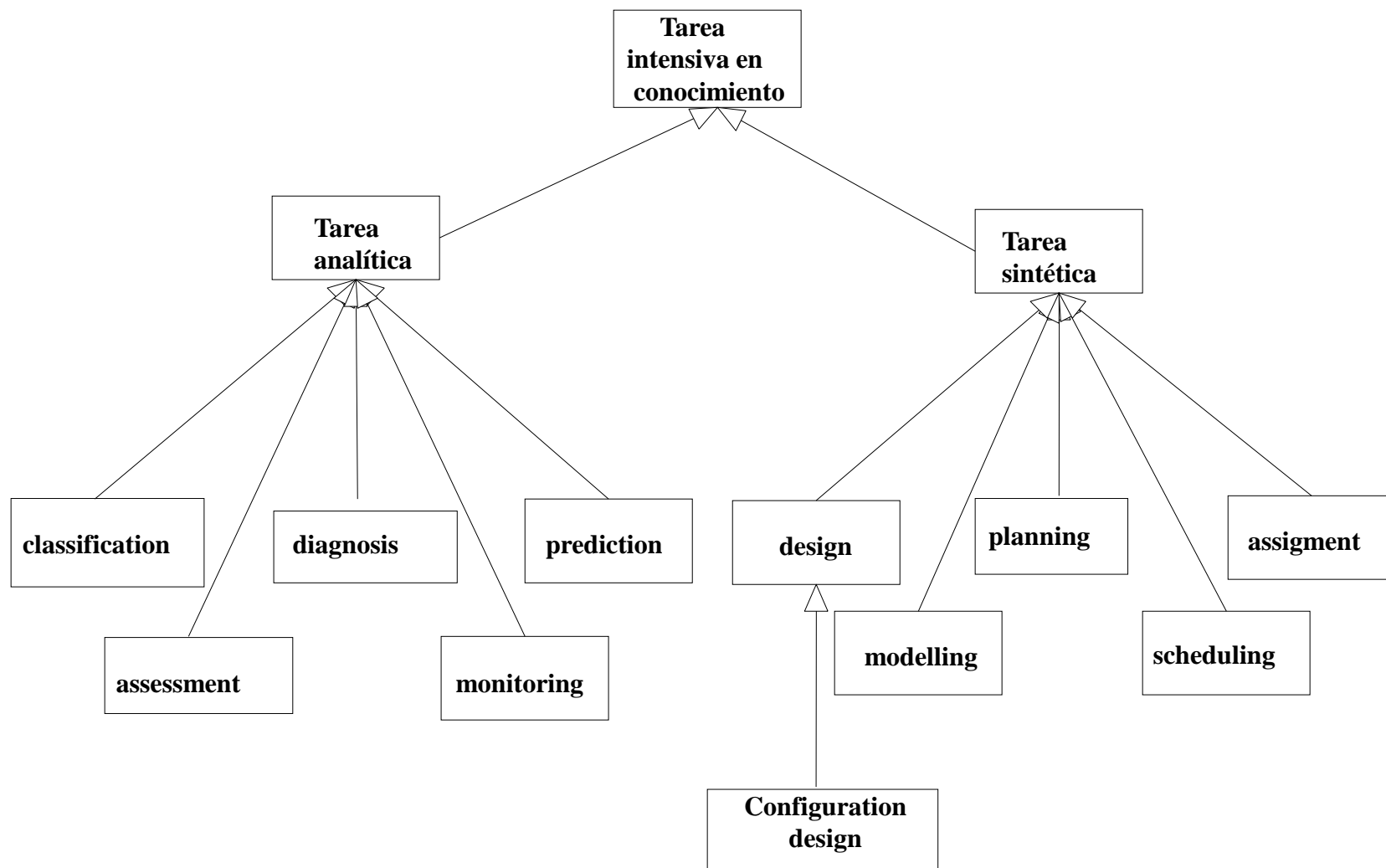
- Los modelos de conocimiento pueden ser parcialmente reutilizados en aplicaciones nuevas.
 - Prevenir “re-inventar la rueda”
 - Aumentar la eficiencia coste/tiempo
 - Disminuir la complejidad
 - **Asegura la calidad del software**

- La guía principal para la reutilización es el tipo de tarea

- Existe un catálogo de plantillas de tareas, que en O-O se denominan “patrones” de tareas intensivas en conocimiento.

- Específica para cada tipo de tarea
- El rango de tipos de tareas está limitado
 - background: ciencia cognitiva y psicología
- La tipología se basa en la noción de “sistema”
- Son combinaciones reutilizables de elementos del modelo de conocimiento
 - estructura inferencial (provisional)
 - estructura de control típica
 - esquema del dominio típico desde el punto de vista de la tarea.
- Soporta el modelado de conocimiento “top-down”

- **Sistema:** Término abstracto para el objeto sobre el que se aplica la tarea.
 - en diagnóstico técnico: artefacto o aparato que está siendo diagnosticado
 - por ej., en la configuración de un ascensor, el ascensor a ser diseñado.
 - No es necesario que exista (aún)
- **Tipos de tareas**
 - **Analíticas**
 - el sistema (la solución) pre-existe
 - aunque típicamente no es completamente “conocido”
 - entrada: algunos datos acerca del sistema
 - salida: alguna caracterización del sistema
 - **Sintéticas**
 - el sistema (la solución) no existe aún
 - entrada: requisitos del sistema que va a construirse
 - salida: descripción del sistema construido



Tipo tarea	Entrada	Salida	Conocimiento	Características
Análisis	Observaciones del sistema	Caracterización del sistema	Modelo del sistema	Descripción del sistema
Classification	Object features	Object class	Asociaciones feature-class	El conjunto de clases está predefinido
Diagnosis	Symptoms/Complaints	Fault category	Modelo de comportamiento del sistema	Varía forma de salida (cadena causal, estado, componente), depende del uso
Assessment	Case description	Decision class	Criterios, normas	Performed at one particular point in time
Monitoring	System data	Discrepancy class	Comportamiento normal del sistema	Sistema cambia en el tiempo. La tarea se repite
Prediction	System data	System state	Modelo de comportamiento	Estado de salida es una descripción sistema en tiempo futuro

Tipos de tareas sintéticas

Tipo tarea	Entrada	Salida	Conocimiento	Características
Síntesis	Requisitos	Estructura del sistema	Elementos, restricciones, preferencias	Necesario generar descripción del sistema
Design	Requirements	Artifact description	Componentes, restricciones, preferencias	Puede incluir diseño creativo de componentes
Configuration Design	Requirements	Artifact description	Componentes, diseño esquelético, restricciones, preferencias	Subtipo de Design con componentes predefinidos
Assignment	Two object sets, requirements	Mapping set 1→set2	Restricciones, preferencias	El mapeo no tiene que ser uno a uno
Planning	Goals, requirements	Action plan	Acciones,,restricciones, preferencias	Acciones parcialmente ordenadas en el tiempo
Scheduling	Job activities, resources, time slots, requirements	Schedule=actividades colocadas en los slots de tiempo de los recursos	Restricciones, preferencias	Su carácter orientado al tiempo la distingue de assignment
Modelling	Requirements	Model	Elementos modelo, plantillas modelos , Restricciones, preferencias	Puede incluir la Síntesis creativa

- Caracterización general
 - Características típicas de la tarea
- Método por defecto
 - Roles, sub-funciones, estructura control, estructura inferencial
- Variaciones típicas
 - cambios/ajustes más usuales o frecuentes
- Esquema típico del conocimiento del dominio
 - asunciones sobre la estructura subyacente del conocimiento del dominio



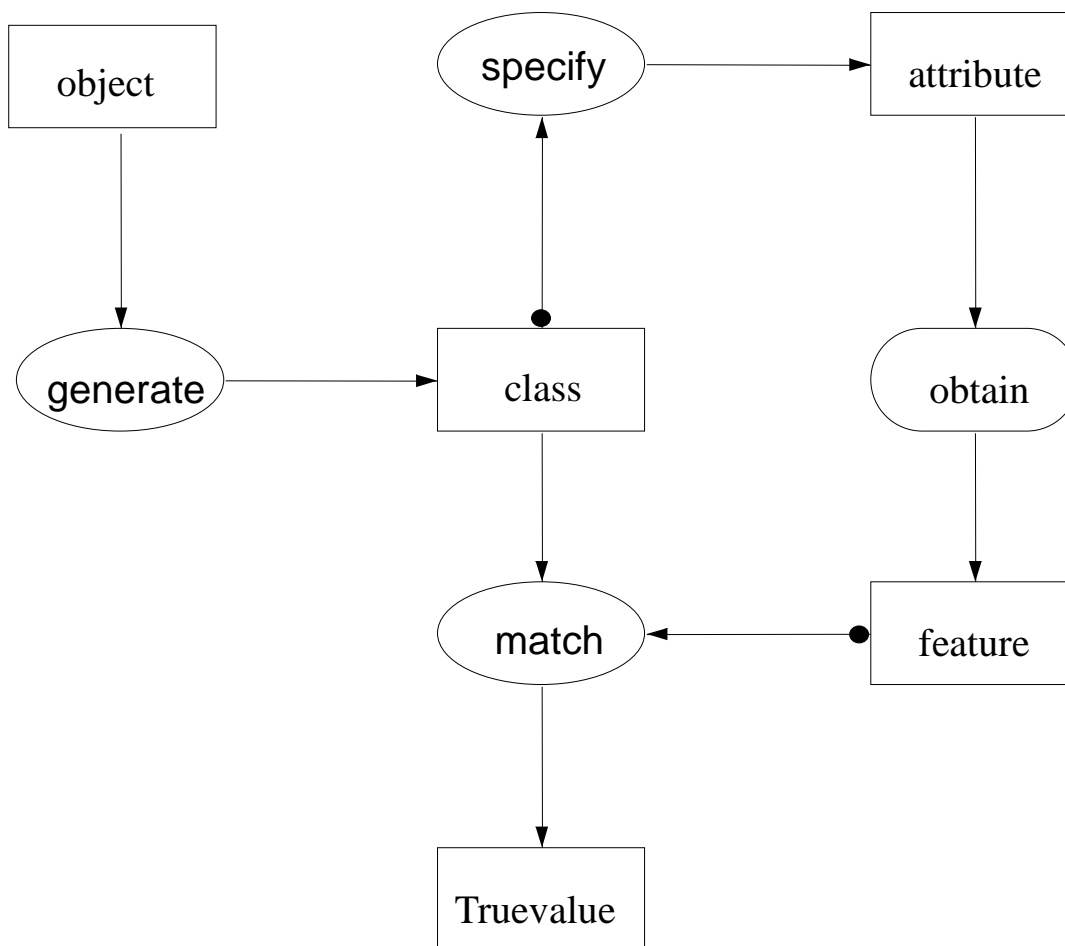
Plantillas de tareas

EJEMPLO DE TAREA ANALÍTICA: Classification

- **Objetivo:** Establecer la clase correcta para un objeto
 - El objeto debe estar disponible para su inspección
 - objetos “naturales”
 - ejemplos típicos: clasific. de rocas, manzanas, setas, peces, ...
- **Es una de las tareas analíticas más simples; muchos métodos**
 - Entrada: el objeto a clasificar
 - Salida: la clase a la que pertenece el objeto
- **Terminología:**
 - object (el objeto que se desea clasificar)
 - class (grupo de objetos similares),
 - attribute (característica observable o inferible del objeto),
 - feature (par atributo-valor de un objeto, p.ej. color=verde)
- **Otras tareas analíticas implican a veces una tarea de clasificación, ej. diagnóstico**

Método por defecto

- Opciones
 - Dirigido por los datos
 - Dirigido por las metas
- Suele funcionar mejor el dirigido por las metas
 - utilizando posteriormente un método de podado.
- Método de podado-del-conjunto-de-candidatos
 - generar todas las clases a las que el objeto pueda pertenecer
 - especificar un atributo del objeto
 - obtener el valor del atributo
 - eliminar todas las clases inconsistentes con ese valor



TASK classification;

ROLES:

INPUT: object: “Object that needs to be classified”;

OUTPUT: candidate-classes: “Classes consistent with the object”;

END TASK classification;

TASK-METHOD prune-candidate-set;

REALIZES: classification;

DECOMPOSITION:

INFERENCES: generate, specify, match;

TRANSFER-FUNCTIONS: obtain;

ROLES:

INTERMEDIATE:

class: “object class”;

attribute: “a descriptor for the object”;

new-feature: “a newly obtained attribute pair”;

current-feature-set: “the collection of features obtained”;

truth-value: “indicates whether the class is consistent with
object features obtained during the reasoning process”;

CONTROL-STRUCTURE:

```

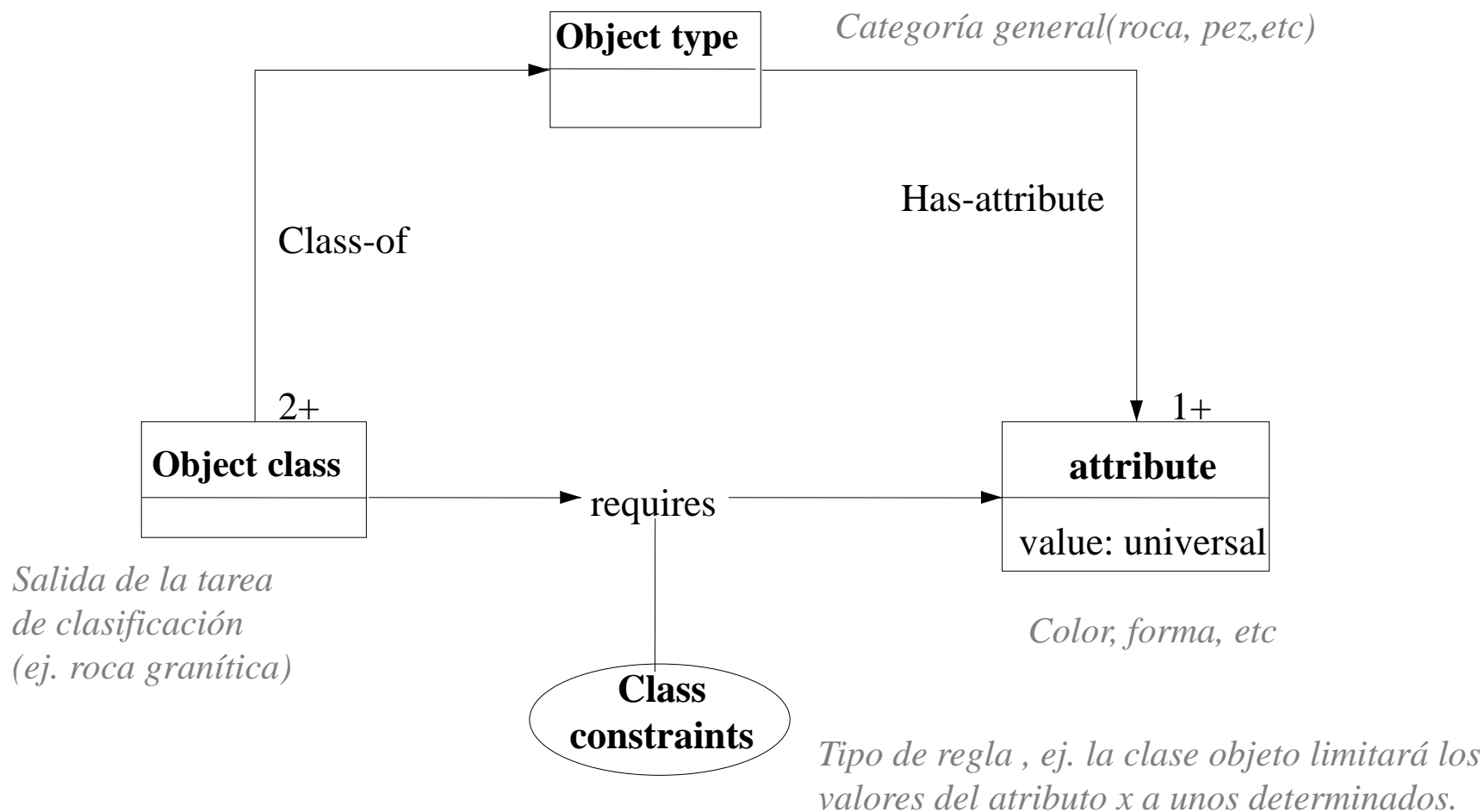
while new-solution generate (object -> class) do
  candidate-classes := class add candidate-classes;
end while
while new-solution specify (candidate-classes -> attribute)
  and size candidate-classes > 1 do
    obtain (attribute -> new-feature);
    current-feature-set := new-feature add current-feature-set;
    for-each class in candidate-classes do
      Match (class + current-feature-set -> truth-value);
      if truth-value == false;
      then candidate-classes := candidate-classes subtract class;
      end-if
    end for-each
  end while
END TASK-METHOD prune-candidate-set;
  
```


- Limitación de generación de candidatos
 - a través de un proceso dirigido por los datos
 - considerando sólo los candidatos consistentes

- Diferentes formas de selección de atributos
 - árbol de decisión
 - teoría de la información
 - control del usuario

- Búsqueda jerárquica a través de la estructura de clases

Esquema del dominio



- monitoring + diagnosis
 - Proceso de producción
- monitoring + assessment
 - Tareas de enfermería
- diagnosis + planning
 - dispositivos de investigación, conflictos
- classification + planning
 - Aplicaciones militares

- El SBC va a construir un horario, que no existe en este momento
 - Tarea de Síntesis.
- Las posibles plantillas son 5:
 - Synthesis.
 - Goal: Dado un conjunto de requisitos construir una estructura de sistema que cumpla estos requisitos.
 - Configuration Design.
 - Goal: Dado un conjunto de componentes predefinidos, encontrar un ensamblaje de éstos que satisfaga los requisitos y las restricciones.
 - Assignment.
 - Goal: Crear una relación entre dos grupos de objetos (subjects y resources) que cumpla los requisitos y restricciones.
 - Planning.
 - Goal: Dada una meta, generar un plan que consista en un conjunto de acciones o actividades ordenadas.
 - Scheduling.
 - Goal: Dado un cjto de trabajos predefinidos, cada uno de los cuales consiste en actividades secuenciadas temporalmente llamadas unidades, asignar todas las unidades a los recursos, satisfaciendo a la vez las restricciones.

- **Synthesis:**
 - Mirando la descripción de la plantilla, en Features dice que se debe usar en tareas en las que el conjunto de estructuras posibles sea limitado.
 - La eliminamos porque como vemos en el método por defecto, es necesario generar todas las posibles soluciones, lo cual en nuestro problema no es abordable.
 - Comenta que otras plantillas, como Configuration design, utiliza métodos inteligentes para limitar el nº de diseños generado.

- **Assignment:**
 - Selecciona un grupo de objetos sujeto que va a asignar a un recurso. Luego considera una localización, que es una relación entre sujeto y recurso.
 - No parece muy adecuado porque no tiene en cuenta el tiempo y hay varios sujetos a asignar (podemos asignar aulas a profesores, profesores a asignaturas, aulas a horas, profesores a horas...

- **Planning:**
 - Dada la descripción de una meta, generar un plan que consista en un conjunto ordenado de acciones o actividades que nos lleven a la meta.
 - La entrada es la meta a lograr con los requisitos necesarios
 - Las plantillas son Synthesis o Synthesis+Configuration Design
 - Si el espacio de planes posibles es grande, es necesario usar Configuration Design.

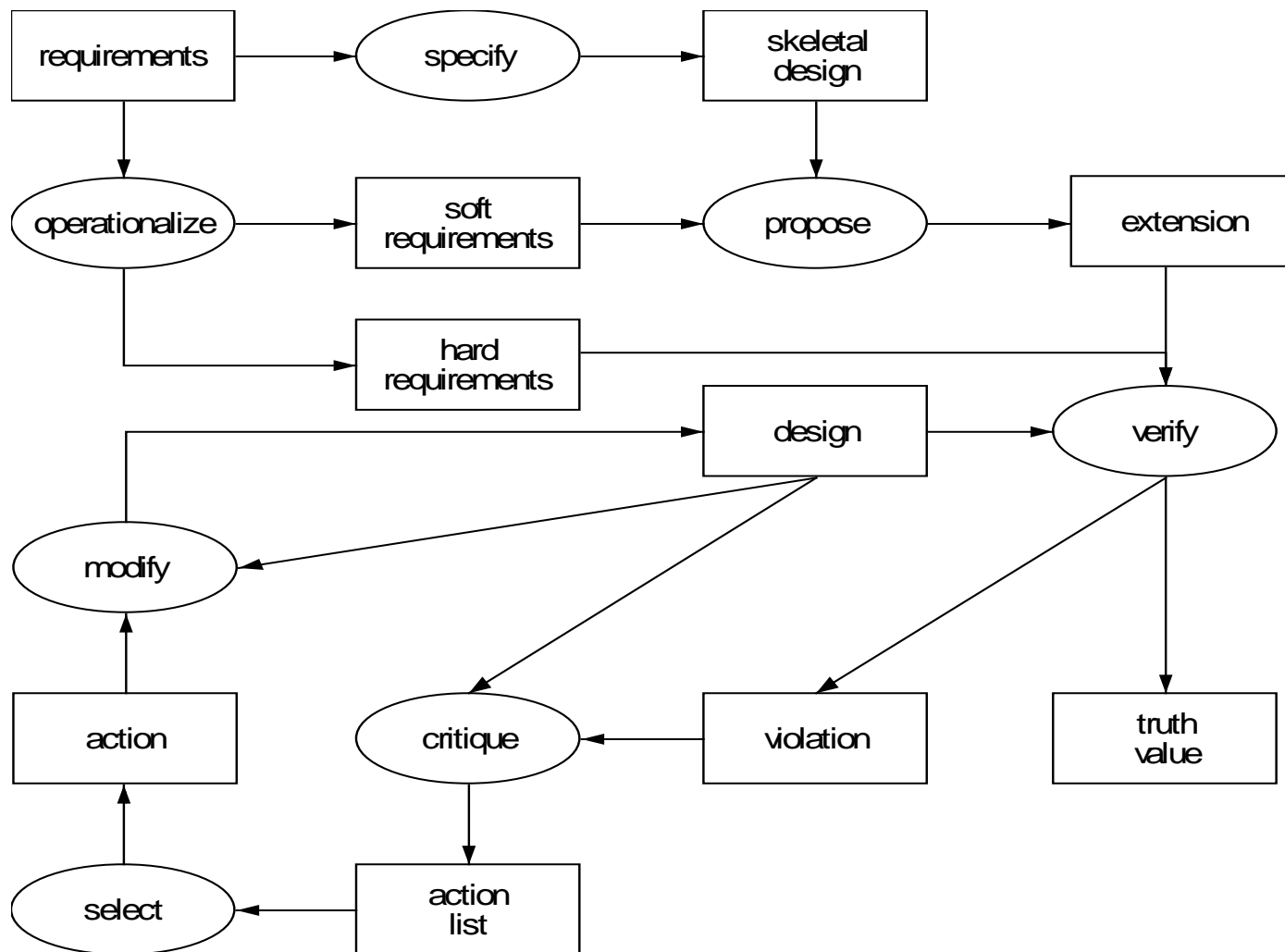
- **Scheduling:**
 - Toman como entrada actividades cuyas secuencias temporales están predefinidas
 - Para crear esas secuencias temporales hay que usar la plantilla de Planning
 - El primer paso de planning no es factible, porque el espacio de posibles soluciones es muy amplio.
- **Configuration Design**
 - La meta es encontrar un ensamblado de componentes que satisfaga los requisitos y obedezca a restricciones.
 - Esto nos permite separar requisitos obligatorios y preferencias de los horarios.
 - La solución se puede construir incrementalmente, verificando los diseños intermedios y corrigiéndolos.
- **Elegimos Configuration Design**

- El sistema a construir es un artefacto físico (Ej. diseño de un vehículo)
- Puede incluir el diseño creativo de los componentes, demasiado complejo para el estado tecnológico actual
- Configuration design: subtipo del diseño que excluye la creatividad
 - Es una forma de diseño automatizable
- Objetivo: Dado un conjunto de componentes predefinido, encontrar una combinación que se ajuste a los requisitos y que verifique las restricciones.
 - Ejemplo: configuración de un ascensor o un PC
- Entrada: Un conjunto de requisitos
- Salida: Un ensamblaje que consiste en componentes y valores de parámetros instanciados.
- Terminología:
 - component, una parte del ensamblaje (ej. Disco duro de un ordenador);
 - parameter, una característica de un componente (ej. Capacidad de almacenamiento de un disco duro), o de un ensamblaje de componentes (ej. Precio total de un ordenador);
 - constraint, restricción en la elección de un componente (ej. La tarjeta de CPU x no puede tener más de y Mb de memoria interna), o del valor de un parámetro (la carga máxima del cable de red es z)
 - preference, permite elegir un diseño particular en el conjunto de diseños válidos. Normalmente suele haber varias soluciones correctas, las preferencias se usan para indicar el diseño preferido entre éstos (por ej. Minimizar el precio)
 - Requirement, necesidades y deseos de los usuarios futuros del sistema a configurar. Los requisitos se traducen de forma típica en constraints y preferences (hard & soft).

Configuration Design: Método por defecto “propose & revise”

- Bucle básico:
 1. Proponer una extensión del diseño
 2. Verificar el nuevo diseño, si Ok continuar con paso 1, si no, ir a paso 3
 3. Criticar el diseño actual y generar una lista ordenada de acciones para revisarlo.
 4. Seleccionar una acción y modificar el diseño hasta que la función de verificación tenga éxito.
 5. Volver al paso 1. Si no hay más extensiones, informar de la configuración final.
- Requisitos específicos del conocimiento del dominio
 - revisar estrategias
- El método también puede usarse para otras tareas sintéticas
 - asignación (assignment) con backtracking
 - planificación esquelética (con estructuras de planes estándar)

Configuration Design: Plantilla inferencial



TASK configuration-design;

ROLES:

INPUT: requirements: “requirements for the design”;

OUTPUT: design: “the resulting design”;

END-TASK configuration-design;

TASK-METHOD propose-and-revise;

REALICES: configuration-design;

DECOMPOSITION:

INFERENCES: operationalize, specify, propose, verify, critique, select, modify;

ROLES:

INTERMEDIATE:

soft-requirements: “requirements to be used as preferences”;

hard-requirements: “requirements that are hard constraints”;

skeletal-design: “set of design elements”;

extension: “a single new value for a design element”;

violation: “constraint violated by the current design”;

truth-value: “boolean indicating result of the verification”;

action-list: “ordered list of possible repair (fix) actions”;

action: “a single repair action”;

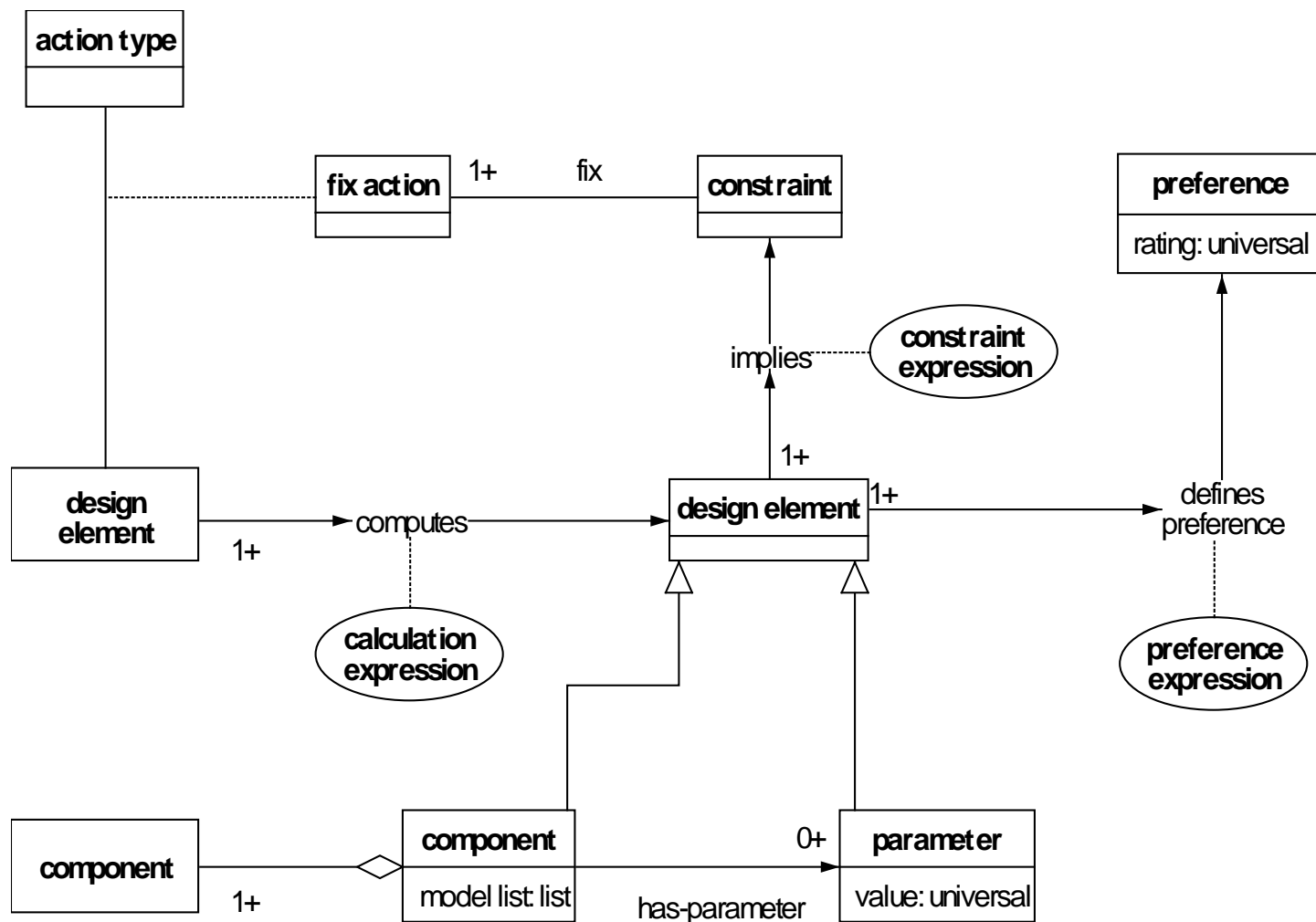
CONTROL-STRUCTURE:

```

  operationalize(requirements -> hard-reqs + soft-reqs);
  specify(requirements -> skeletal-design);
while new-solution propose(skeletal-design + design +soft-reqs -> extension) do
  design := extension add design;
  verify(design + hard-reqs -> truth-value + violation);
  if truth-value == false then
    critique(violation + design -> action-list);
    repeat
      select(action-list -> action);
    modify(design + action -> design);
    verify(design + hard-reqs -> truth-value + violation);
    until truth-value == true;
    end-repeat
  End-if
end while
END TASK-METHOD propose-and-revise;
  
```

1. Realizar una verificación y una revisión sólo cuando se ha propuesto un valor para todos los elementos del diseño
 - Requiere un ajuste sencillo en la estructura de control del método
 - Puede tener un impacto importante en la competencia del método.

2. Evitar el uso de conocimiento de reparación
 - Las reparaciones son heurísticas de búsqueda que nos guían en el espacio de diseños alternativos, que potencialmente es grande, una vez que se viola una restricción.
 - Los ajustes pueden no estar disponibles o bien estarlo parcialmente
 - alternativa: backtracking cronológico para el proceso de revisión
 - La solución es computacionalmente más costosa



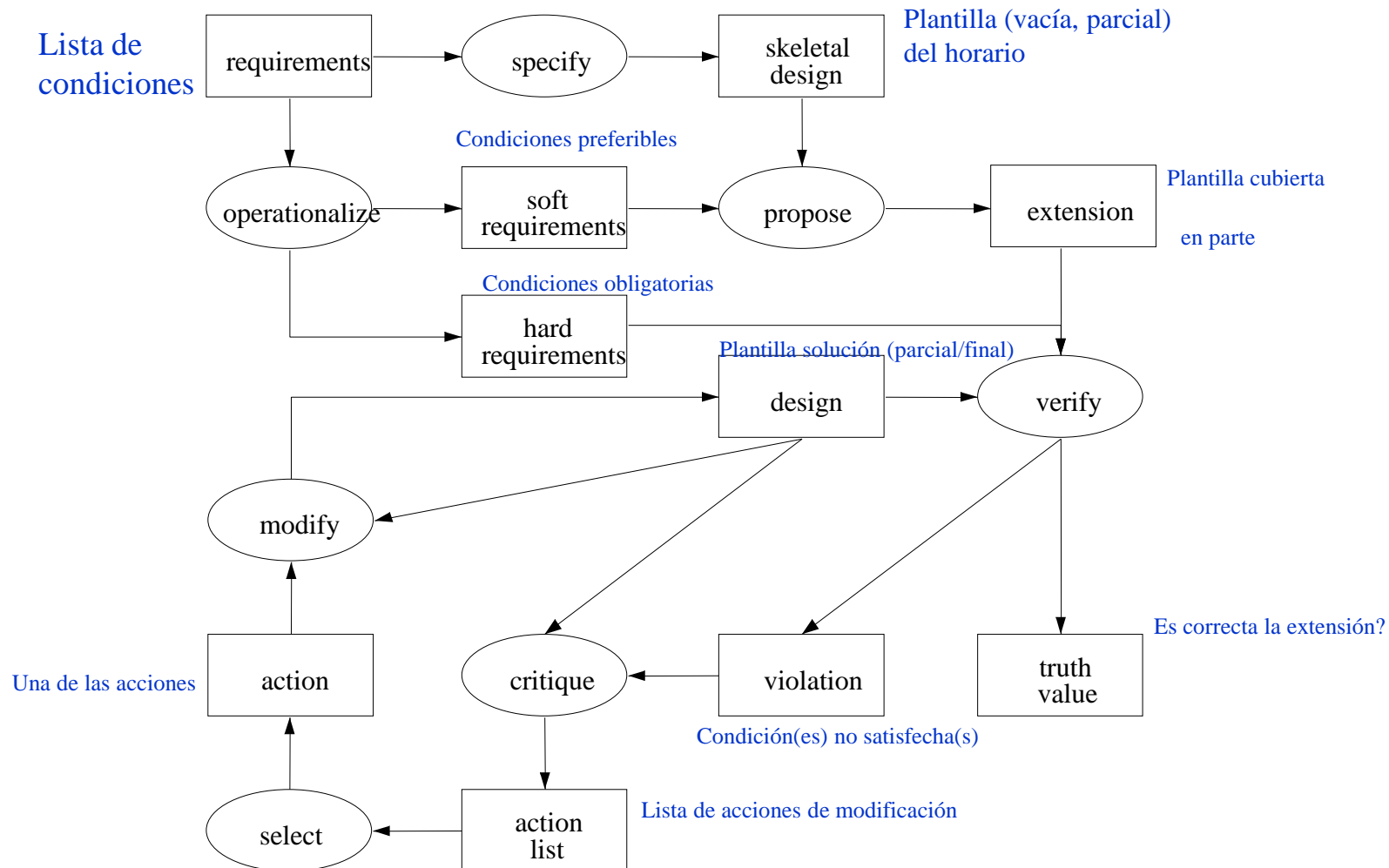
Ejemplo de horarios. Tarea y Método Tarea

```

TASK-METHOD propose-and-revise;
  ■ REALICES: configuration-design;
  ■ DESCOMPOSITION:
    • INFERENCES: operationalize, specify, propose, verify, critique, select, modify;
    • TRANSFER-FUNCTIONS: obtain1, obtain2;
  ■ ROLES:
    • INTERMEDIATE:
      ○ soft-requirements: "preferencias: requisitos que sería deseable cumplir";
      ○ hard-requirements: "restricciones: requisitos que hay que cumplir";
      ○ extension: "diseño con algún valor nuevo";
      ○ violation: "restricción violada por el diseño";
      ○ truth-value: "booleano que indica el resultado de la verificación";
      ○ action-list: "lista de las posibles reparaciones del diseño";
      ○ action: "una modificación";
      ○ continue-value: "booleano que indica si se quiere buscar otra solución"
  ■ CONTROL-STRUCTURE:
    operationalize(requirements → hard-requirements + soft-requirements);
    specify(requirements + skeletal-design → skeletal-design);
    continue-value:=true;
    WHILE continue-values == true AND NEW-SOLUTION propose (skeletal-design + design + soft
    soft-requirements → extension) DO

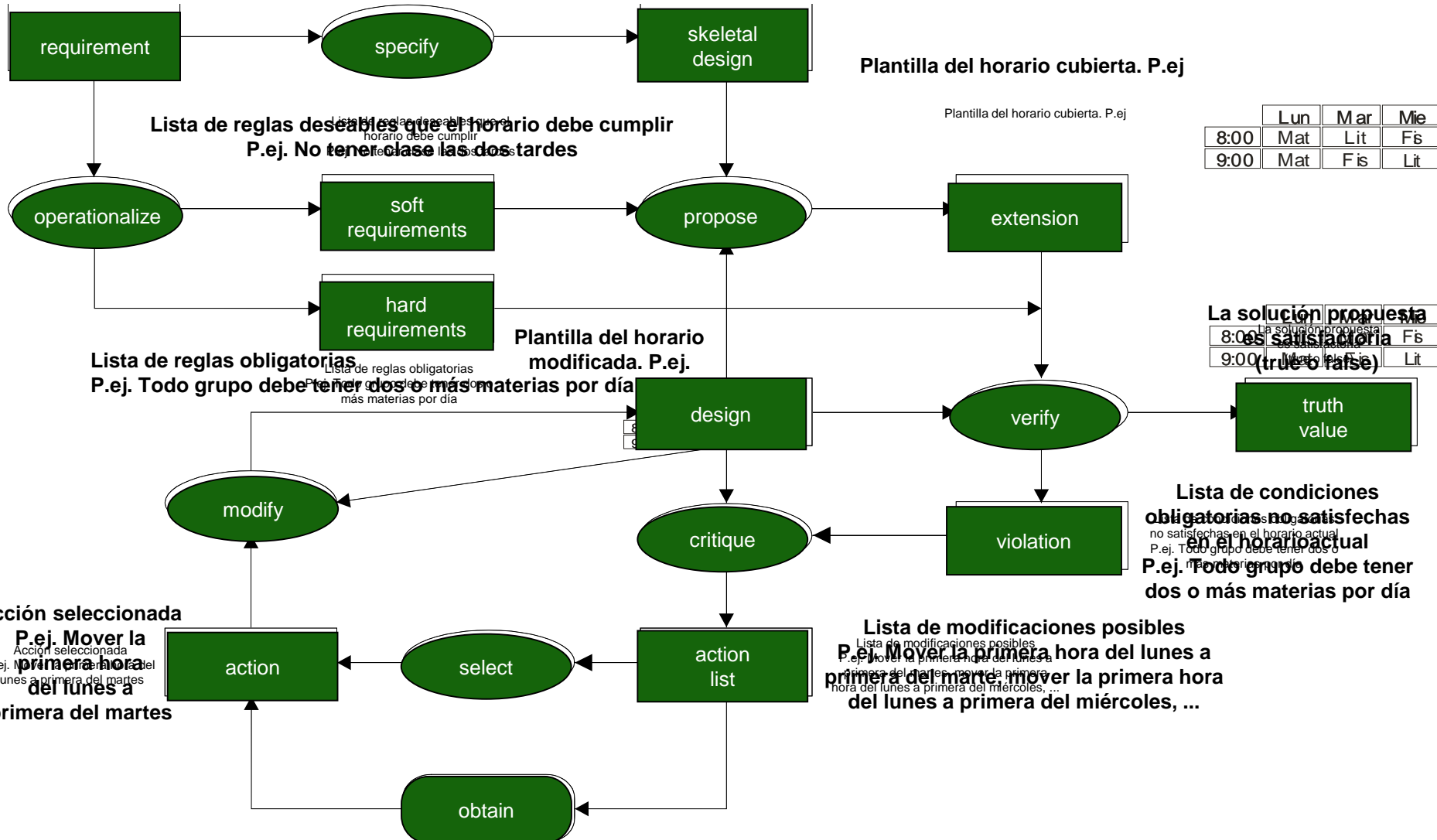
      design:=extension ADD design;
      verify(design + hard-requirements → truth-value+violation);
      WHILE truth-value == false DO

        critique(violation + design → action-list);
        select(action-list → action);
        obtain1(action + action-list → action);
        modify(design + action → design);
        verify(design + hard-requirements → truth-value + violation);
      END WHILE
      obtain2(design → continue-value);
    END WHILE
END TASK-METHOD
  
```



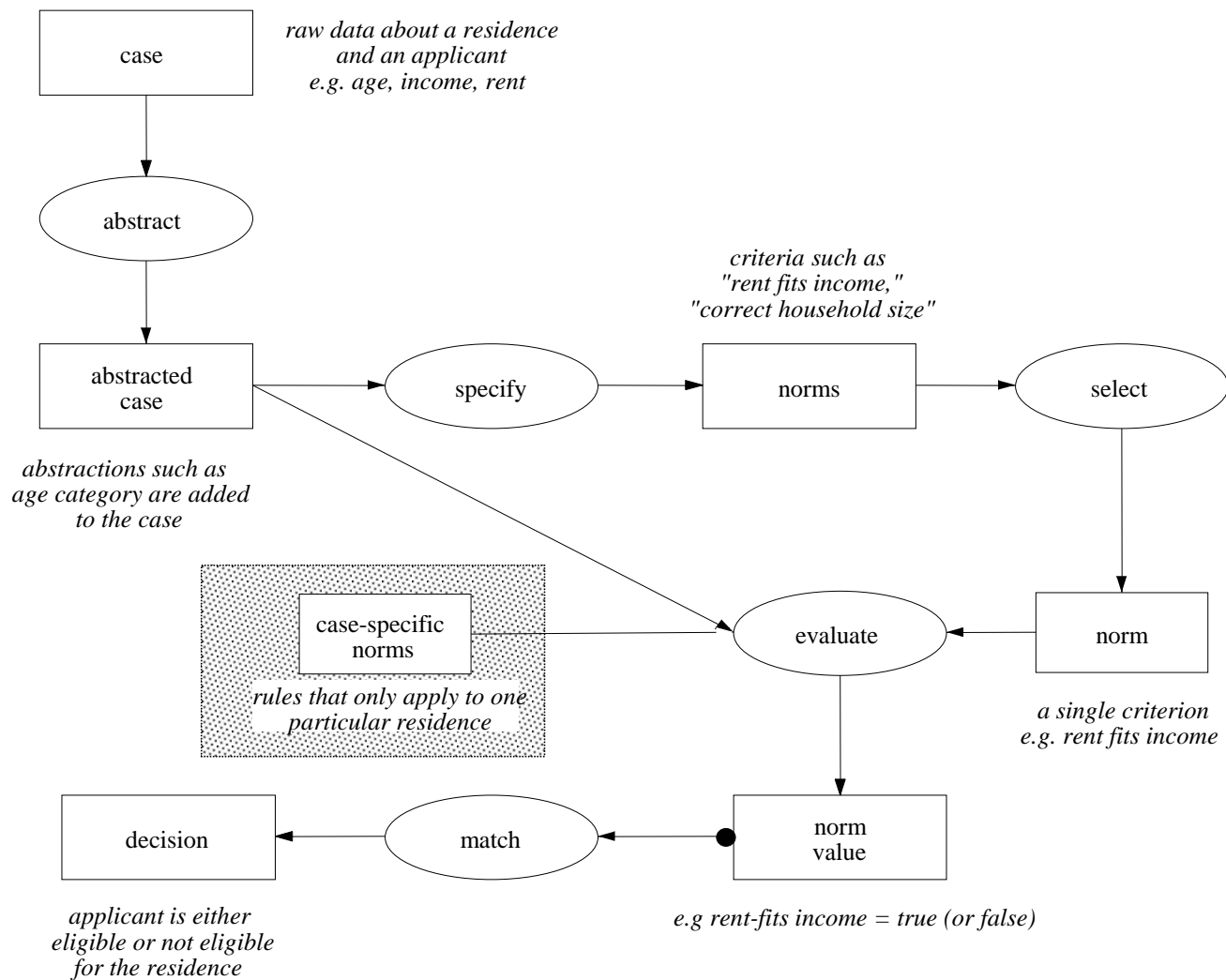
Plantilla inicial del horario. P.ej.

		Partida inicial del horario		
		Lun	Mar	Mie
8:00		8:00		
9:00		9:00		



- El problema consiste en saber si una persona (solicitante) es elegible o no para una vivienda social.
- En principio, debe ser una tarea analítica.
 - Clasificación: Podría ser
 - Assessment: Encontrar una decisión para un caso basándose en una serie de normas del dominio. Podría ser.
 - Diagnosis: En principio, la entrada es “complaint”, hay observables intermedios,... No parece adecuada.
 - Monitoring: No parece adecuada. La entrada son datos que se recogen cada cierto tiempo.
 - Al analizar la estructura de control de clasificación, parece poco adecuada, ya que hay que generar la clase primero (en nuestro caso sería elegible/no elegible) y especificar atributos. Si algún atributo no se cumple, la clase no concuerda, lo que no es así en nuestro dominio.
 - Finalmente, elegimos la plantilla de assessment.

Plantilla assessment: Estructura inferencial anotada para Housing

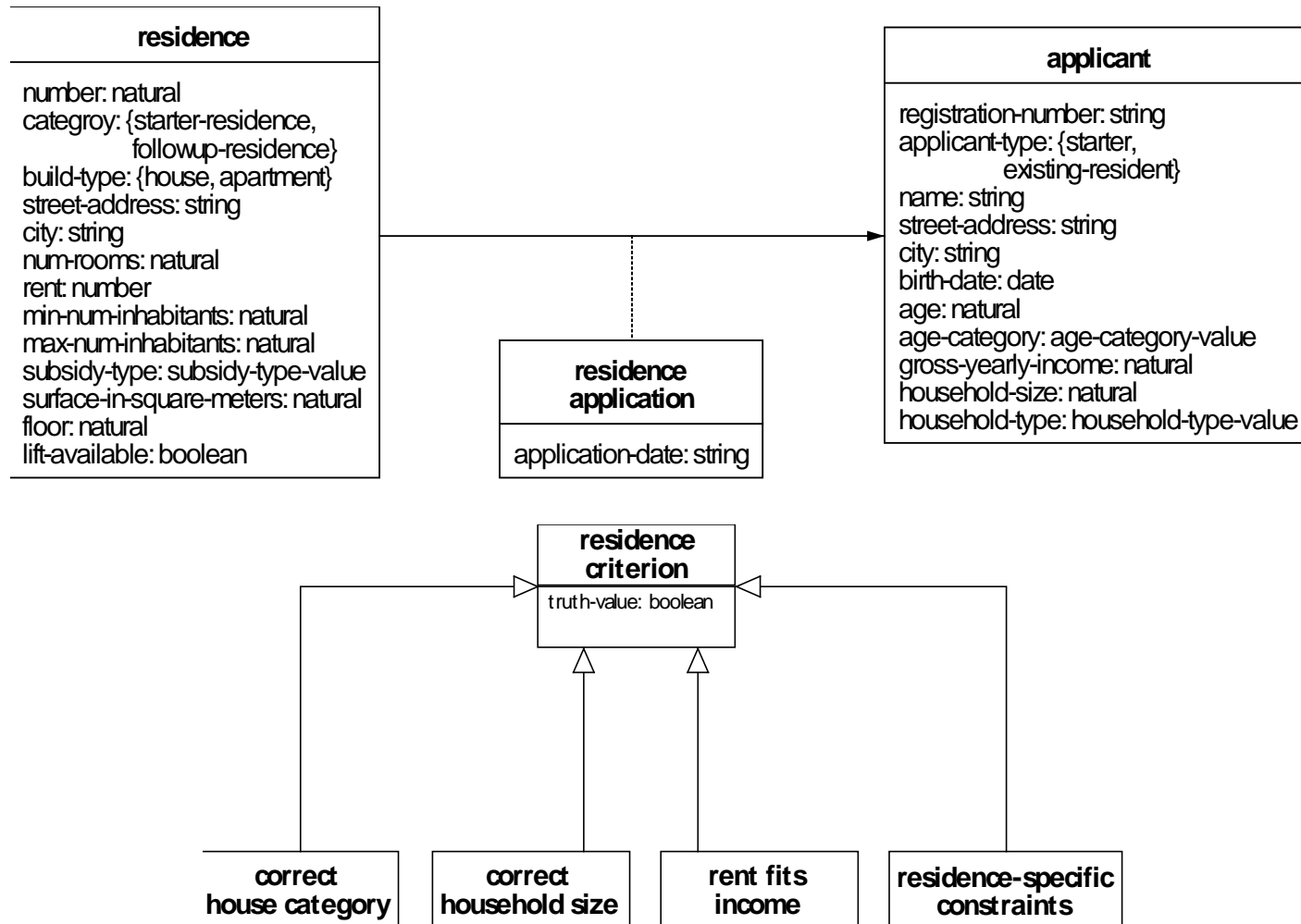


Método control assessment

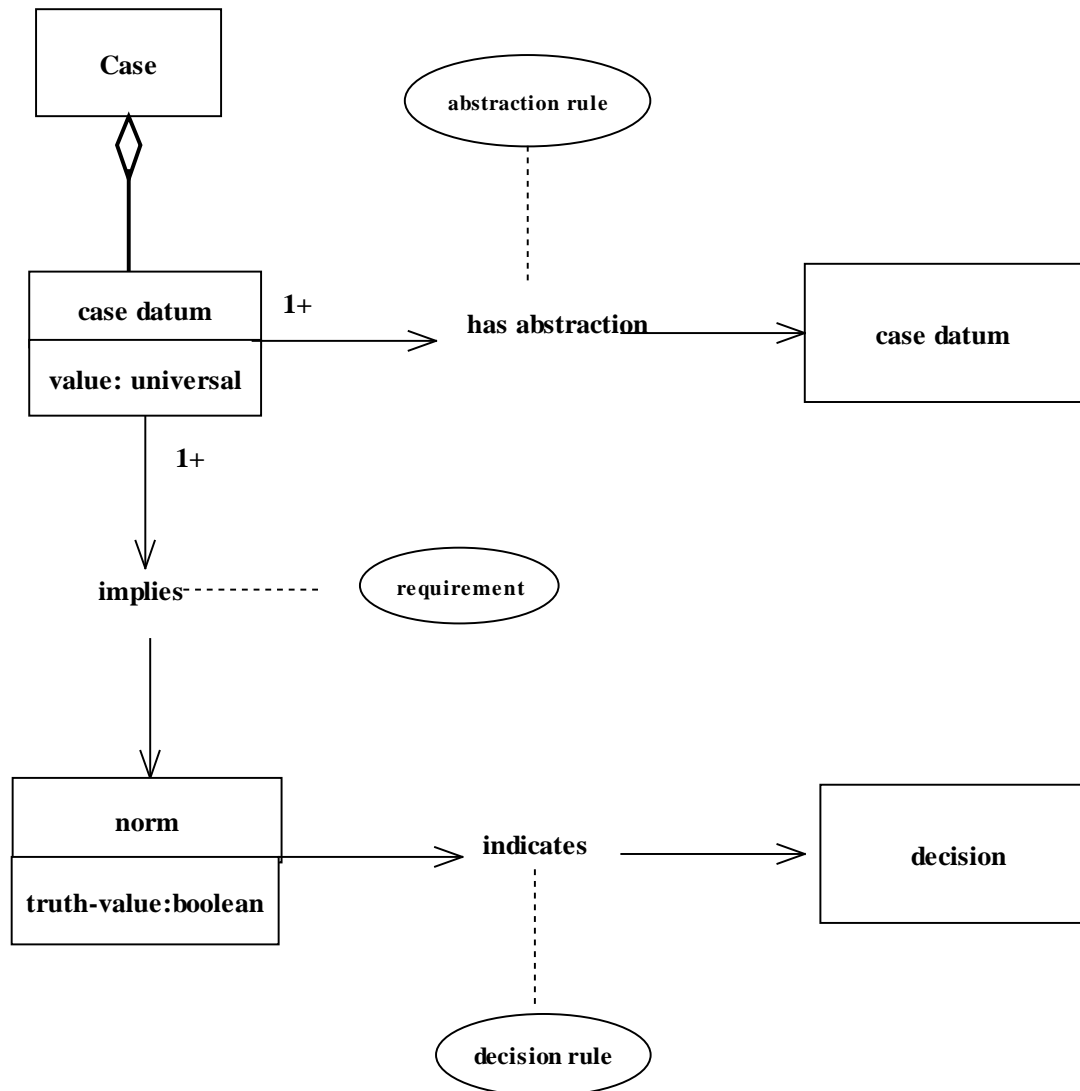
- TASK assessment;
 - ROLES:
 - INPUT: case-description: “The case to be assessed”;
 - OUTPUT: decision: “The result of assessing the case”;
 - END TASK assessment;
 - TASK-METHOD assessment-with-abstraction;
 - REALICES: assessment;
 - DECOMPOSITION:
 - INFERENCES: abstract, specify, select, evaluate, match;
 - ROLES:
 - INTERMEDIATE:
 - abstracted-case: “The raw data plus the abstractions”;
 - norms: “The full set of assessment norms”;
 - norm: “ A single assessment norm”;
 - norm-value: “”truth value of norm for this case”;
 - evaluation-results: “List of evaluated norms”;
 - CONTROL-STRUCTURE:
 - WHILE
 - HAS-SOLUTION abstract (case-description ---> abstracted case)
 - DO
 - case-description:= abstracted-case;
 - END-WHILE
 - specify (abstracted-case→norms);
 - REPEAT
 - select (norms → norm);
 - evaluate (abstracted-case+norm → norm-value);
 - evaluation-results:= norm-value ADD evalutaion-results;
 - UNTIL
 - HAS- SOLUTION match (evaluation-results → decision);
 - END REPEAT
 - END TASK-METHOD aseesment-with-abstraction;

- Actividad 2.1.- Selección de la plantilla de tareas.
- Actividad 2.2.- Especificación inicial del esquema de conocimiento del dominio.
 - El esquema del dominio se recupera de la plantilla elegida
 - La descripción de conceptos, relaciones y subtipos es similar a la construcción de un modelo de objetos inicial (sin métodos) en OO.
 - Se lleva en paralelo con la selección de la plantilla
 - Es más data-oriented que knowledge-oriented
- Actividad 2.3.- Completar la especificación del modelo de conocimiento.

Esquema inicial del problema Housing



Esquema del dominio de la plantilla

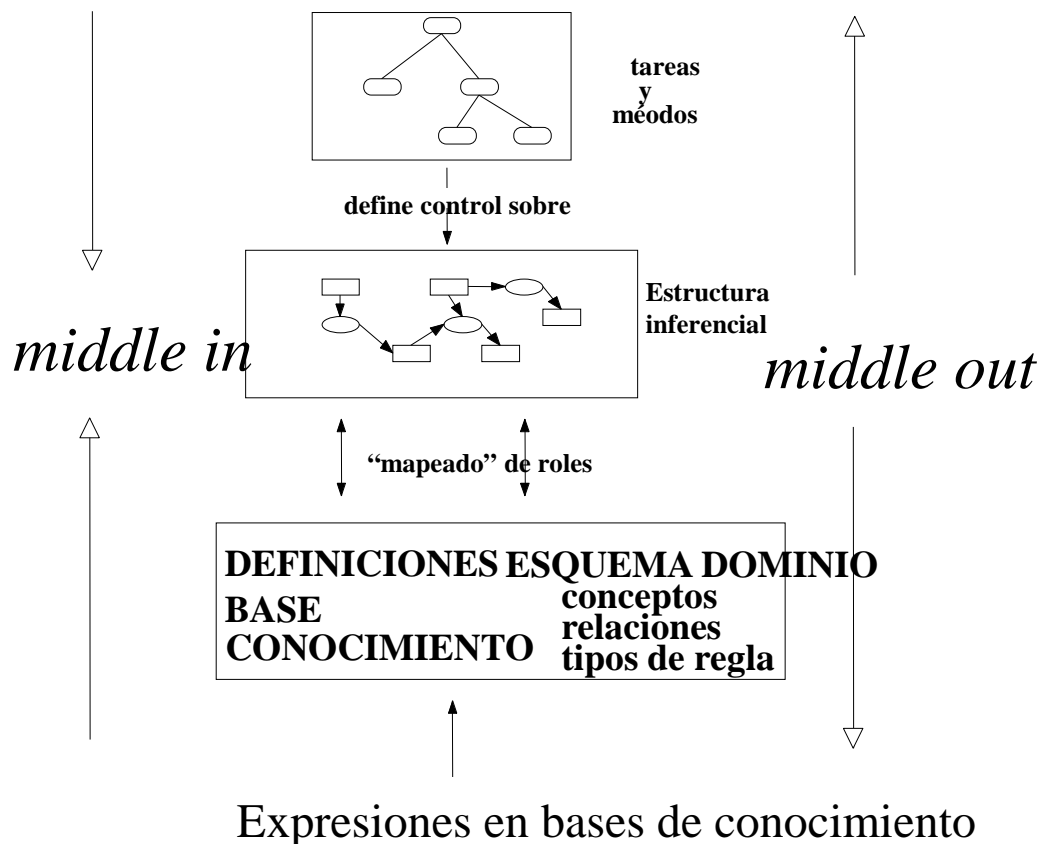


- Actividad 2.1.- Selección de la plantilla de tareas.
- Actividad 2.2.- Especificación inicial del esquema de conocimiento del dominio.
- **Actividad 2.3.- Completar la especificación del modelo de conocimiento.**
 - Opción 1: Middle-out
 - Empezar con el conocimiento inferencial
 - Es la aproximación preferida
 - Precondición: la plantilla de la tarea proporciona una buena aproximación de la estructura inferencial.
 - Opción 2: Middle-in
 - Comenzar en paralelo con la descomposición de la tarea y el modelado del dominio.
 - Consume más tiempo
 - Es necesario si la plantilla de la tarea es demasiado “grano grueso”

Middle-in y Middle-out

La estructura inferencial está suficientemente detallada si:

- la explicación que proporciona lo está.
- es fácil encontrar para cada inferencia un tipo de conocimiento del dominio que actúe como rol estático para ella.





- Técnica de adquisición de conocimiento: Análisis de protocolos
 - Proporciona buenos datos acerca de la estructura del proceso de razonamiento: tareas, control de la tarea, inferencias.
 - La adecuación de la plantilla de la tarea se puede asesorar usándola como un revestimiento de la transcripción de un protocolo
 - La idea es que el IC debería ser capaz de interpretar todos los pasos de razonamiento que hace el experto en el protocolo en términos de una tarea o una inferencia en la plantilla

- Cuando empezemos a especificar un TASK-METHOD, empezar con la CONTROL-STRUCTURE.
- Olvidar detalles de la memoria de trabajo (diseño) , centrarse en caracterizar la estrategia de razonamiento
 - Ej: orden de las tareas, realizar la inferencia hasta que no se produzcan más soluciones, etc.
- Escoger nombres de roles que claramente indiquen el rol.
 - "modelar es nombrar"
- No incluir roles de conocimiento estáticos como parte de entrada/salida de la tarea.
 - Cuando describimos inferencias
- Aplicaciones en tiempo real:
 - se requiere control asíncrono
 - diagramas de transición de estados (preferible)

- Comenzar con la representación gráfica
 - tiene menos información que CML, pero es más transparente
- Elegir con cuidado los nombres de los roles
 - Independencia del dominio
 - Carácter dinámico
 - hipótesis, datos iniciales, hechos
 - saber si son uno sólo o un conjunto, p.ej. la inferencia select acepta como entrada sólo un conjunto.
- Usar en lo posible un conjunto estándar de inferencias
 - consultar el catálogo de inferencias en el libro

- No es diferente del modelado estándar de datos
 - Uso de las técnicas conocidas: entrevistas, análisis de textos, etc.
- Los tipos de conocimiento del dominio que se usen como roles estáticos no necesitan tener la representación “exacta” que se necesita para una inferencia.
 - es un tema de diseño
 - punto clave: saber que el conocimiento está disponible.
- El ámbito del conocimiento del dominio suele ser más amplio que lo que cubren las inferencias
 - En parte este tipo de conocimiento es independiente del proceso de razonamiento (Importante para reutilización)
 - Los tipos de conoc. Dominio no son directamente relevantes para los métodos de la tarea.
 - El modelo de comunicación puede requerir conocimiento del dominio adicional: requisitos de comunicación, explicación, etc.



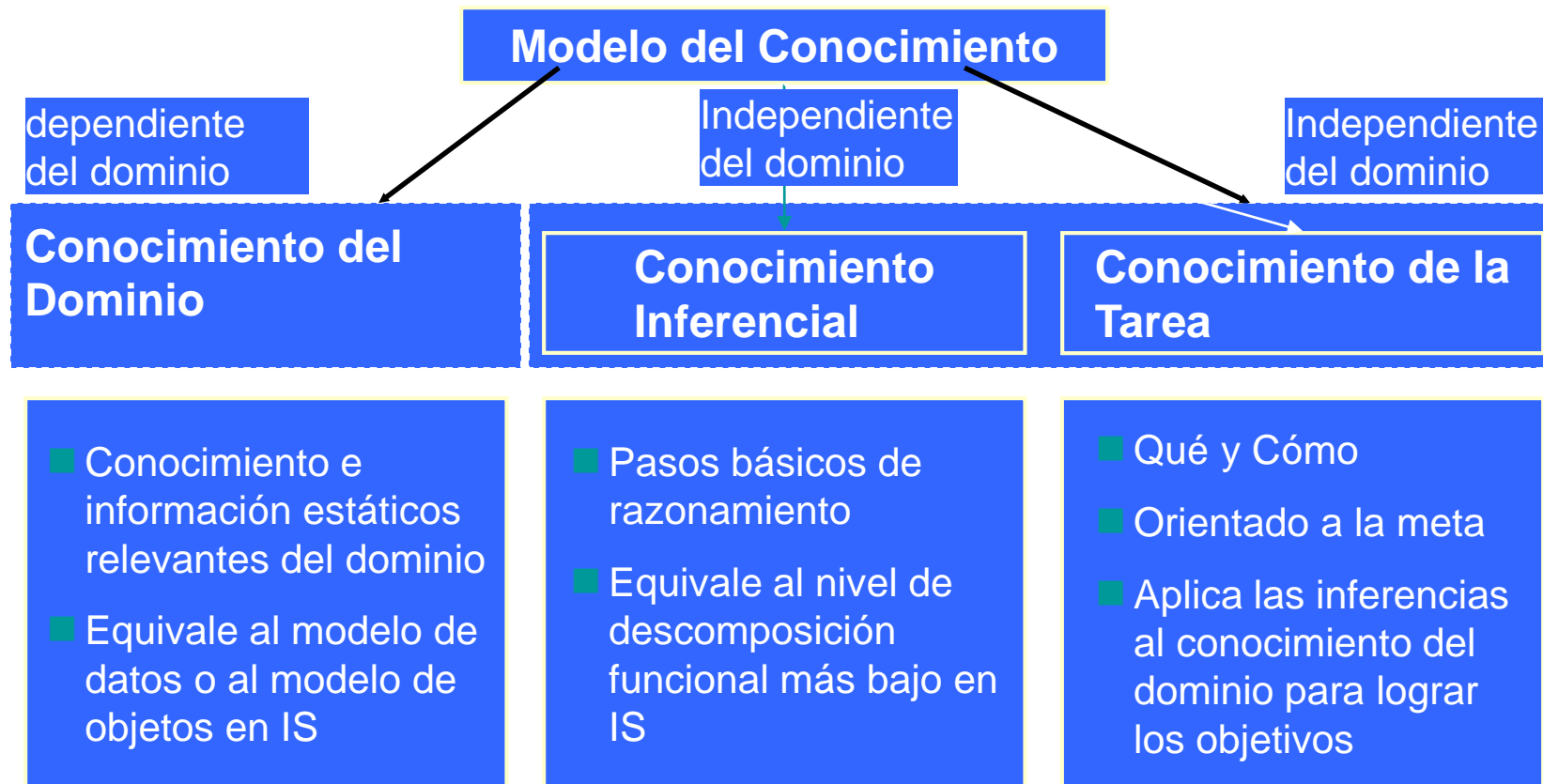
UNIVERSIDADE DA CORUÑA



EL MODELO DEL CONOCIMIENTO. TAREAS, INFERENCIAS y DOMINIO

Desarrollo de Sistemas Inteligentes.

El Modelo de Conocimiento en CommonKADS





UNIVERSIDADE DA CORUÑA

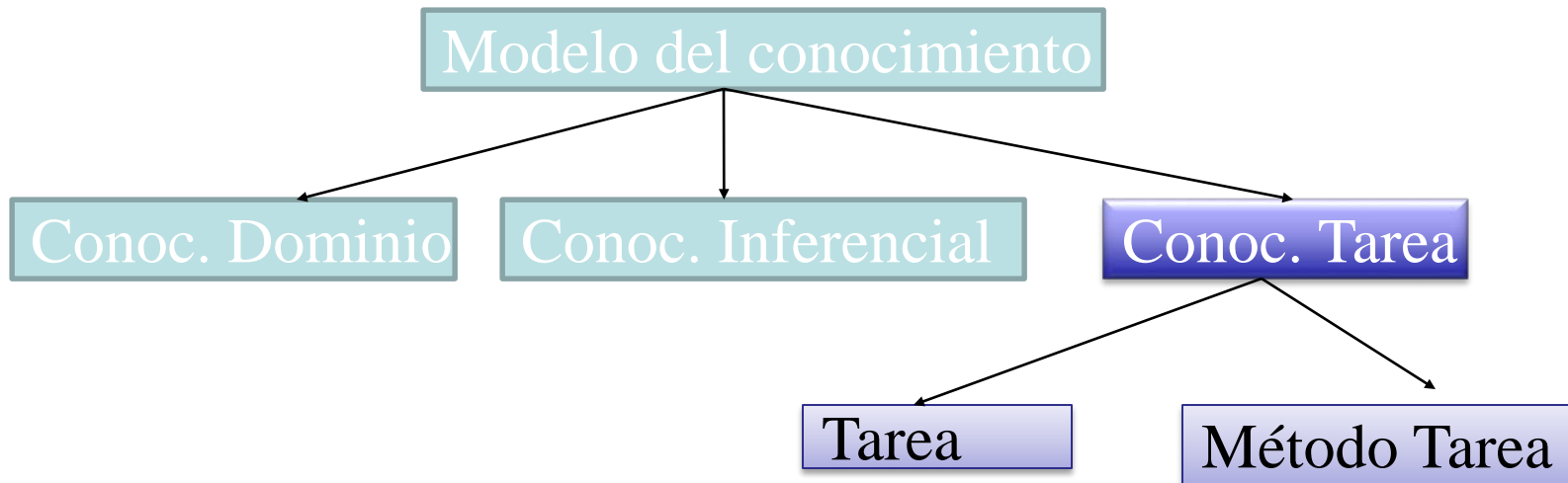


EL MODELO DEL CONOCIMIENTO.

Conocimiento de la Tarea

Desarrollo de Sistemas Inteligentes.

Construcciones del Modelo de conocimiento



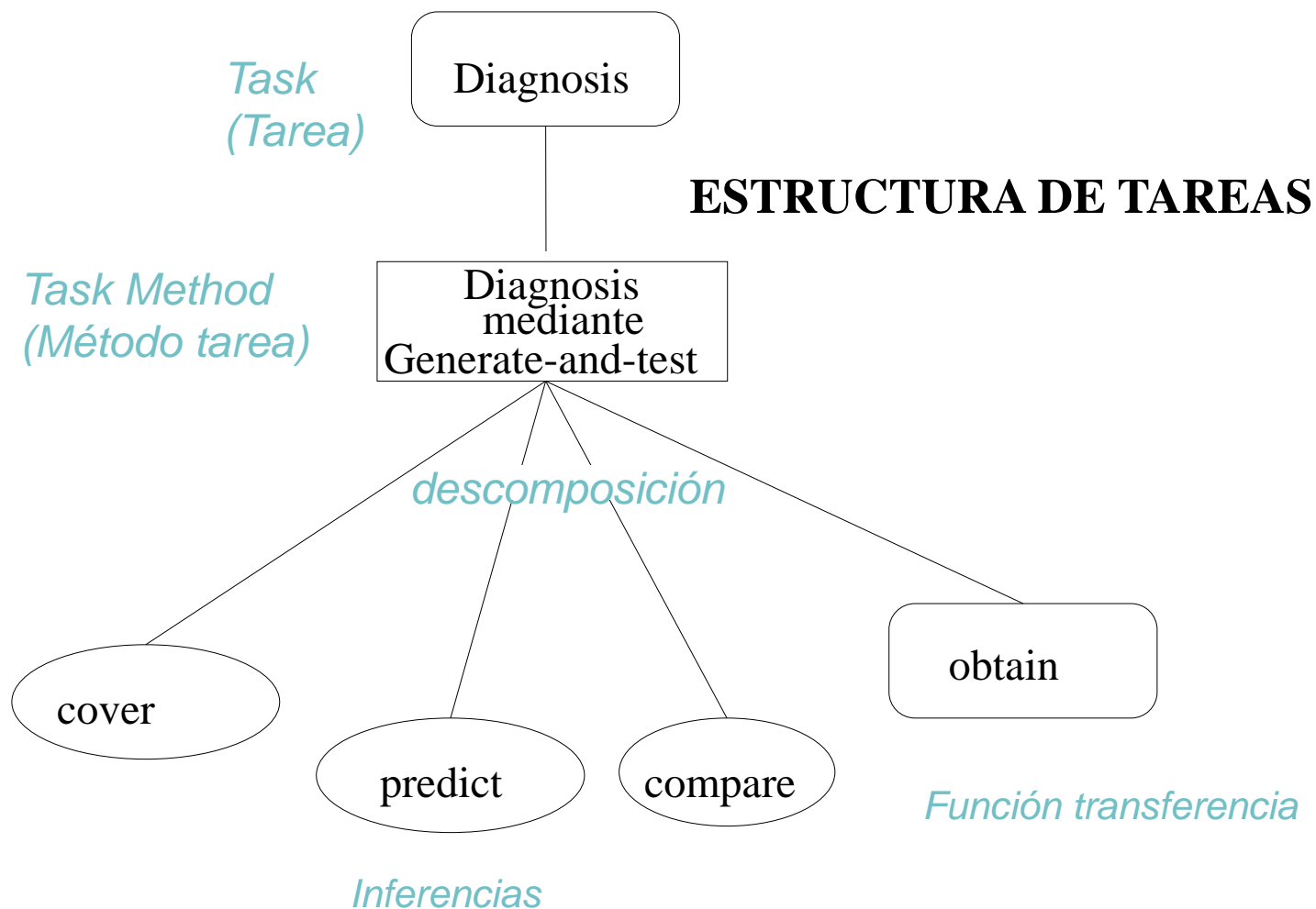


- Describe metas
- Describe las estrategias que se pueden utilizar para realizar esas metas
- Se describe normalmente en modo jerárquico.
 - Tareas-Métodos-Subtareas
- Tiene dos partes:
 - La Tarea (**TASK**) y
 - El Método de la tarea (**TASK-METHOD**).



- Tarea (**TASK**):
 - Define el objetivo del proceso de razonamiento a modelar en función de las entradas y salidas.
 - Los datos manipulados por la tarea se describen independientemente del dominio.
 - Por ejemplo, la salida de una tarea de diagnóstico médico no sería una “enfermedad”, sino un nombre abstracto, como “categoría de avería”

- Método de la tarea (**TASK-METHOD**):
 - Define la estrategia
 - Especificación de su descomposición en tareas, inferencias y funciones de transferencia.
 - Especificación del control.
 - Descripción independiente del dominio.



- Descripción independiente del dominio:
Posibilidad de Reutilización del software
- Slot GOAL: descripción textual informal de la meta de la tarea
- Slot SPECIFICATIONS, descripción textual informal de la relación entre entrada y salida
- Slot ROLES
 - La I/O se especifica en forma de roles funcionales, como en las inferencias.
Diferencias:
 - no hay roles estáticos.
 - no hay mapeado de los roles en términos específicos del dominio.
 - Los roles de las tareas tienen links a los roles inferenciales a través de la estructura de control.
- Cada tarea tiene un método asociado: describe cómo se realiza ésta en términos de subtareas y/o inferencias

TASK Diagnosis;

GOAL: "Encontrar la causa más probable que responda a la queja del usuario";

ROLES:

INPUT:

complaint: "Queja acerca del comportamiento del sistema";

OUTPUT:

fault-category: "Una hipótesis que explique las evidencias encontradas";

evidence: "Conjunto de observaciones que se obtienen durante el proceso diagnóstico";

SPECIFICATIONS: "Encontrar un estado inicial que explique la queja y que sea consistente con las evidencias que se han encontrado";

END_TASK Diagnosis;

TASK-METHOD diagnosis-trough-generate-and-test;

REALIZES: Diagnosis;

DESCOMPOSITION:

INFERENCES: cover, predict, compare;

TRANSFER-FUNCTIONS: obtain;

ROLES:

INTERMEDIATE:

hypothesis: "Una solución candidato" ;

expected-finding: "El hecho que se predice, en el caso de que la hipótesis sea cierta";

actual-finding: "El hecho que se observa";

result: "El resultado de la comparación"

CONTROL-STRUCTURE:

WHILE NEW-SOLUTION

cover (complaint -> hypothesis);

DO

predict (hypothesis -> expected-finding);

obtain (expected-finding -> actual-finding);

evidence := evidence ADD actual-finding;

compare (expected-finding + actual-finding -> result);

IF result== equal;

THEN "break from loop";

END IF

END WHILE

IF result== equal;

THEN fault-category:= hypothesis;

ELSE "no solution found";

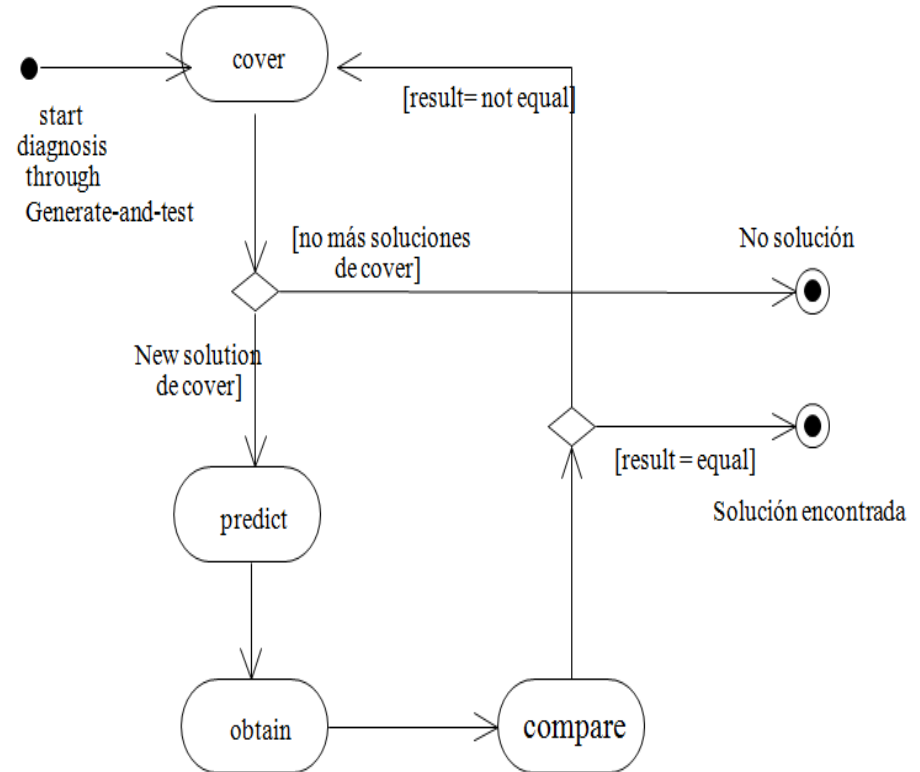
END IF

END TASK-METHOD diagnosis-trough-generate-and-test;

- Descomposición de la tarea en subfunciones.
 - Subfunciones : otra tarea, inferencias o funciones de transferencia.
- Cómo usarlas para alcanzar el objetivo
- Parte central del método: "estructura de control" (CONTROL- STRUCTURE)
 - describe el orden de las subfunciones
 - captura la estrategia de razonamiento que se emplea para resolver el problema
 - pequeño programa,
 - subfunciones=procedimientos y
 - roles= parámetros de los procedimientos
 - puede definir roles de tareas adicionales, que se usan para almacenar resultados temporales

CONTROL-STRUCTURE: Pseudocódigo vs Diagrama de actividad UML

- Llamadas a “procedimientos”
 - Invocación de tareas, inferencias, funciones de transferencia
- Operaciones de datos sobre valores de roles
 - Asignar un valor a un rol, sumar/añadir, restar/borrar, recuperar, ..
- Primitivas de control
 - repeat-until, while-do, foreach-do, if-then-else
- Condiciones:
 - expresiones lógicas sobre roles:
 - **until** diferencial = vacío
 - dos condiciones especiales:
 - Has-solution
 - invocación de inferencia que puede fallar
 - New-solution
 - invocación de inferencia que puede tener éxito varias veces





UNIVERSIDADE DA CORUÑA

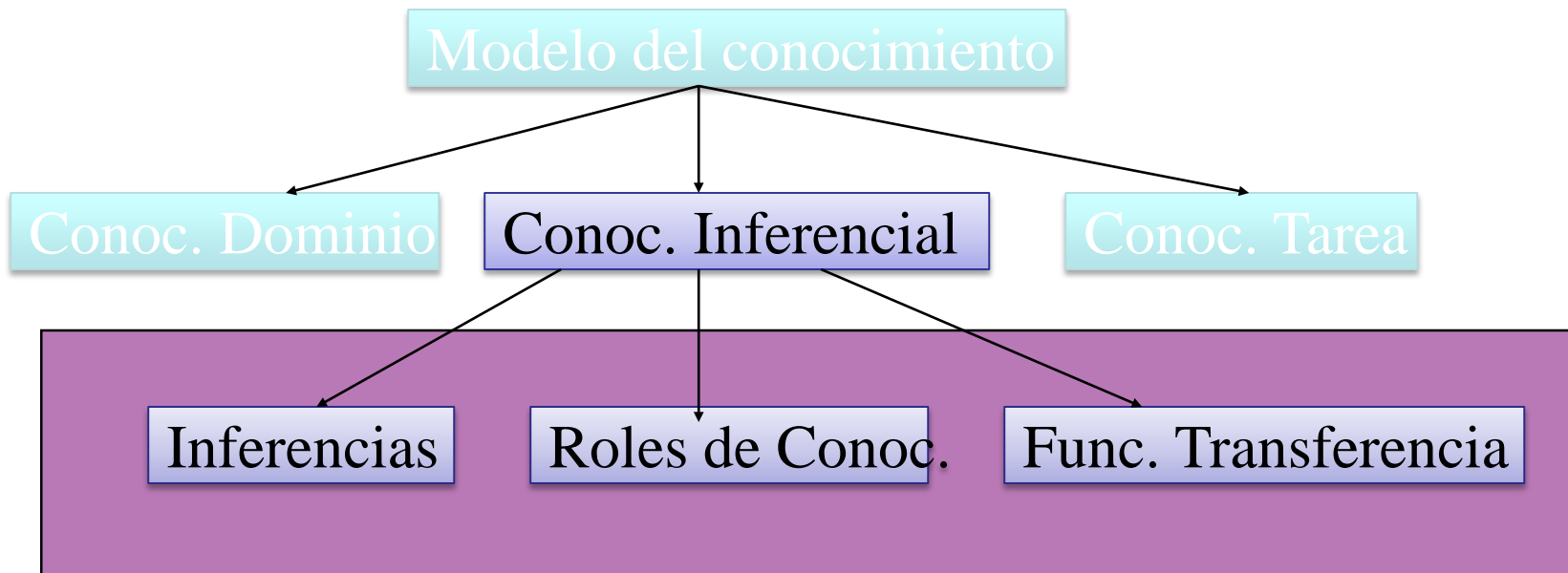


EL MODELO DEL CONOCIMIENTO.

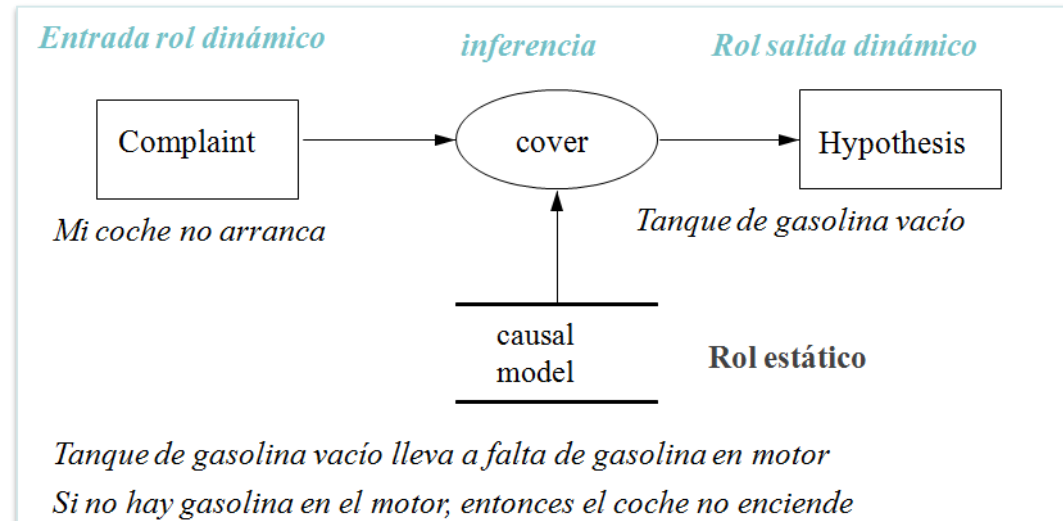
Conocimiento Inferencial

Desarrollo de Sistemas Inteligentes.

Modelo del conocimiento: Conocimiento Inferencial



- Nivel inferior de descomposición funcional
- Describe cómo usar las estructuras estáticas del conocimiento del dominio para realizar el proceso de razonamiento
- Permite la reutilización del conocimiento
- Los elementos principales son:
 - **Inferencias:** Razonamiento. Unidades básicas de procesamiento de información
 - **Funciones de transferencia:** comunicación con otros agentes
 - **Roles de conocimiento:** relación indirecta con el conocimiento del dominio



- Los procesos internos de la inferencia son una caja negra
 - carencia de interés para el modelado de conocimiento.
- Descripción: especificación declarativa de propiedades de E/S.
 - E/S descrita mediante roles funcionales.
 - Nombres abstractos que indican el rol en el proceso de razonamiento.
- Se llaman **roles de conocimiento**
 - **Dinámicos**
 - E/S de la inferencia en tiempo de ejecución.
 - **Estáticos**
 - son estables en el tiempo
 - especifican la colección de conocimiento del dominio que se usa para poder realizar la inferencia.

INFERENCE cover;

ROLES:

INPUT: complaint;

OUTPUT: hypothesis;

STATIC: causal-model;

SPECIFICATION:

”Cada vez que se invoca la inferencia, se genera un candidato solución que puede haber causado la queja. La salida debe ser un estado inicial en el circuito de dependencia de estados que debe “explicar” causalmente la queja de entrada”;

END INFERENCE cover;

- Nombre funcional para elementos dato/conocimiento.
- El nombre captura el rol del elemento en el proceso de razonamiento.
- Mapeado explícito a los tipos del dominio
 - Rol dinámico: variantes E/S
 - Rol estático: entrada invariante
 - una base de conocimiento

```

KNOWLEDGE-ROL complaint;
  TYPE: DYNAMIC;
  DOMAIN-MAPPING: visible-car-state;
END KNOWLEDGE-ROL complaint;

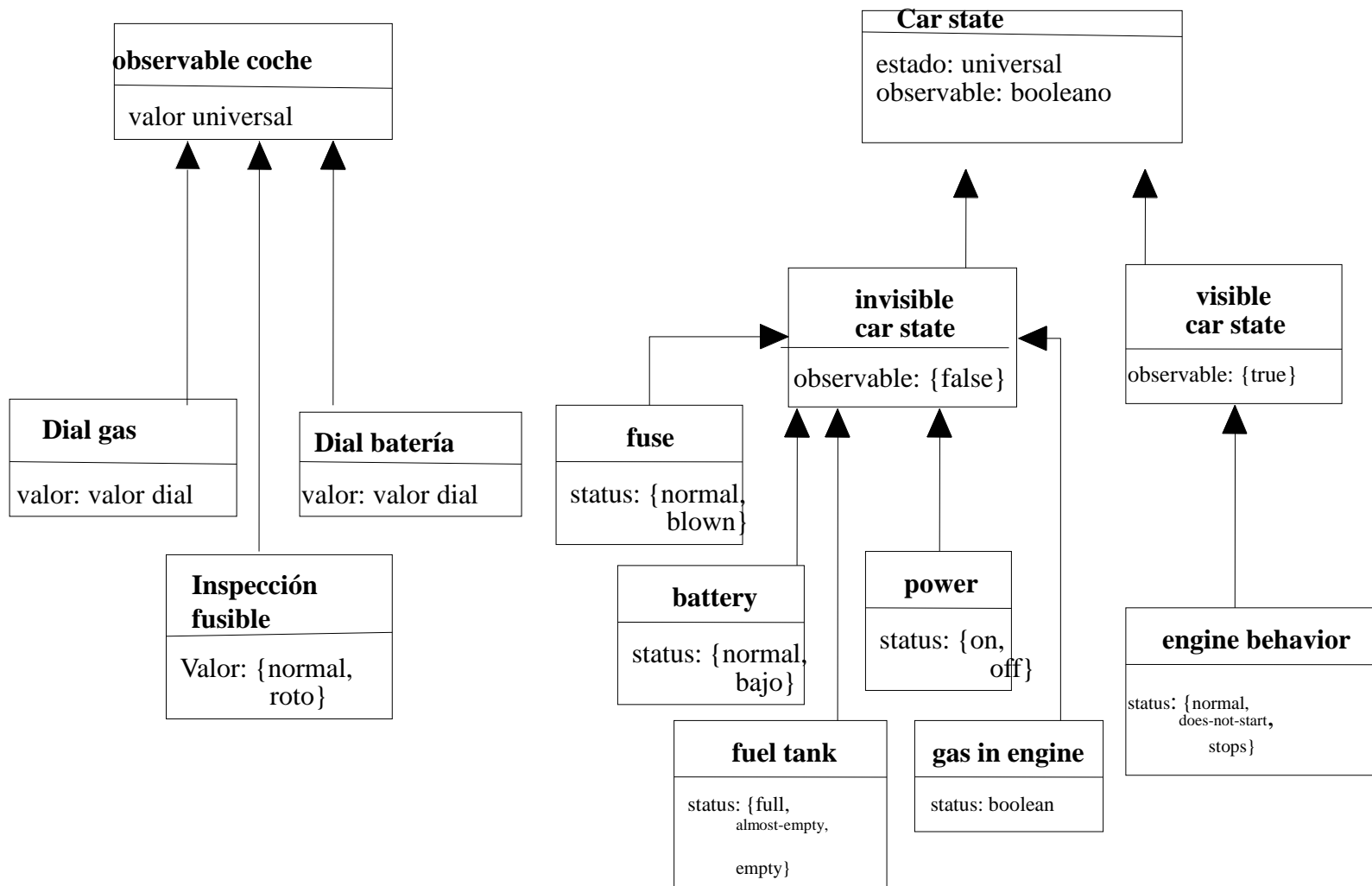
KNOWLEDGE-ROL hypothesis;
  TYPE: DYNAMIC;
  DOMAIN-MAPPING : invisible-car-state;
END KNOWLEDGE-ROL hypothesis;

```

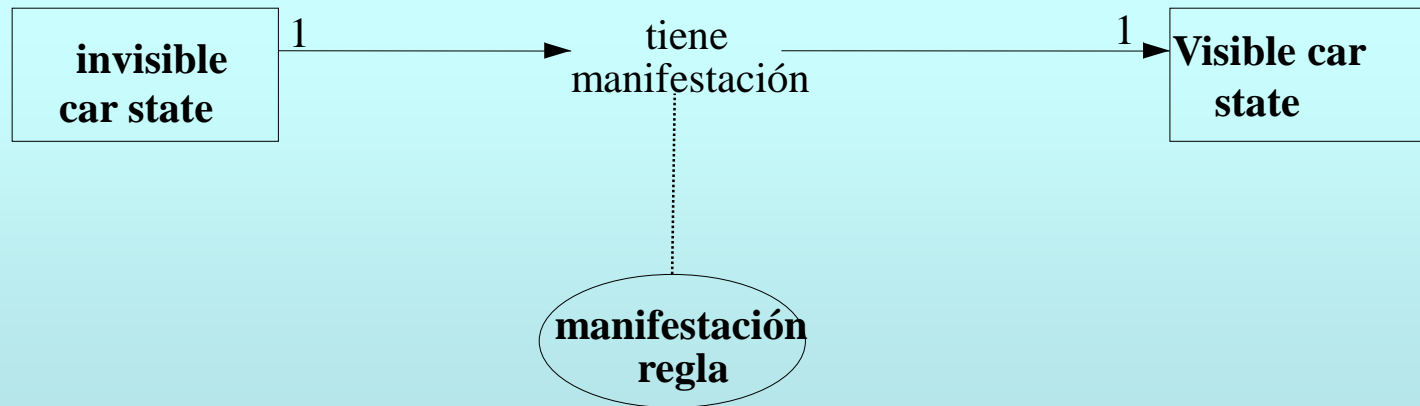
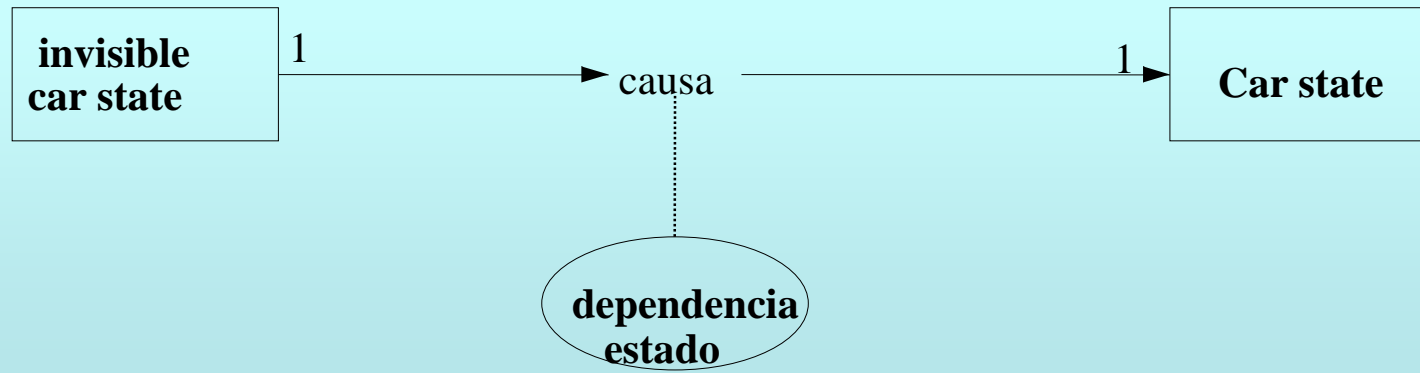
```

KNOWLEDGE-ROL modelo- causal;
  TYPE:STATIC;
  DOMAIN-MAPPING: dependencia-estado FROM circuito-coche;
END KNOWLEDGE-ROL modelo-causal;

```

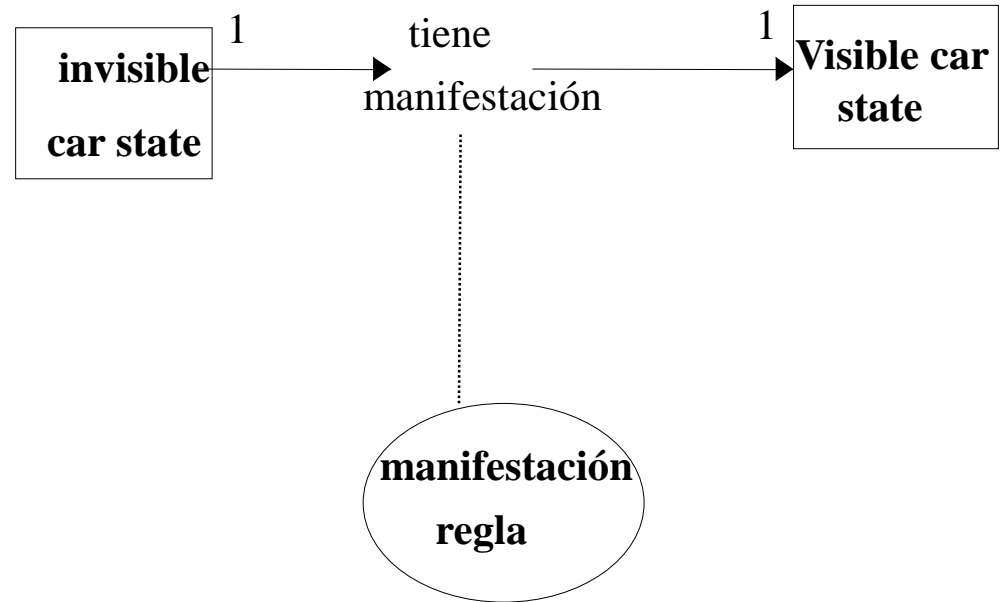


Ejemplo diagnóstico averías coche : Tipos de reglas



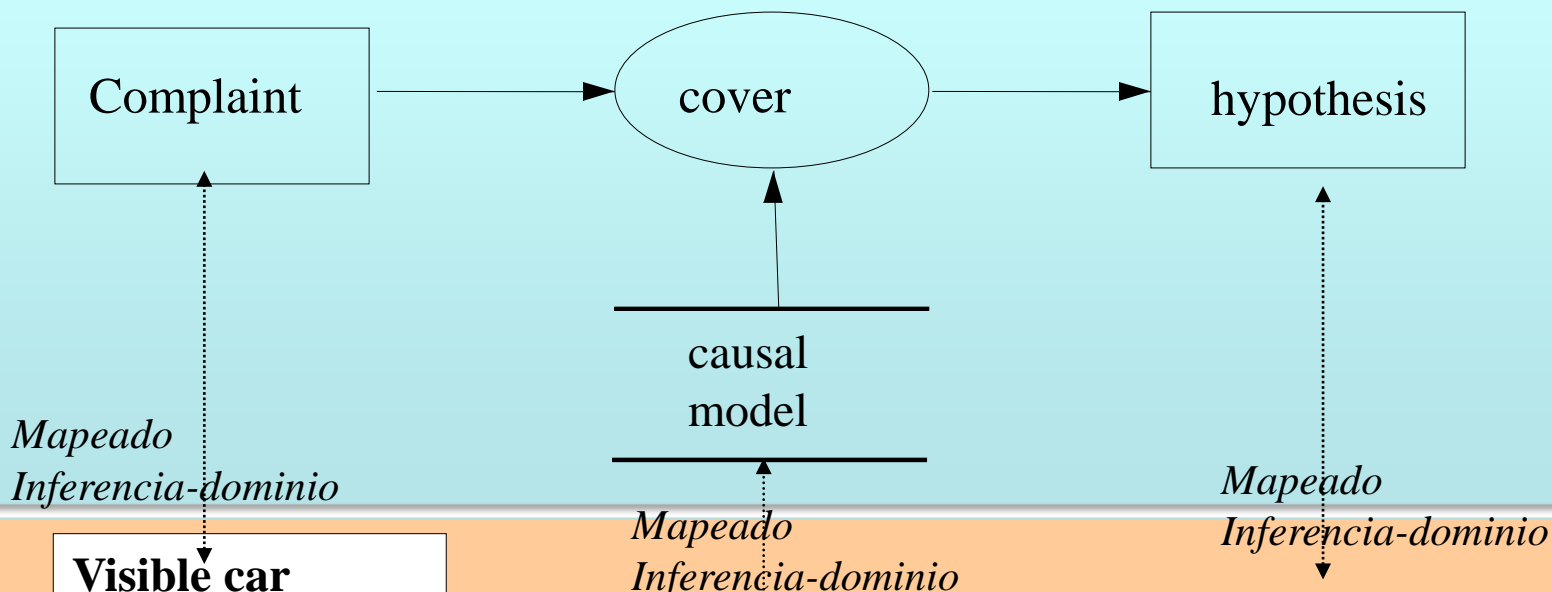
Ejemplo diagnóstico averías coche : Tipo de reglas. Modo texto

RULE_TYPE manifestación- regla;
 DESCRIPTION: “Regla que establece una relación entre un estado interno y un comportamiento externo en términos de un valor observable”;
 ANTECEDENT:
 invisible-car-state;
 CONSEQUENT:
 Visible-car-state;
 CONNECTION-SYMBOL:
 tiene-manifestación;
 END TIPO-DE-REGLA manifestación-
 regla;



Ejemplo diagnóstico averías coche : Aplicación inferencias-dominio

Conoc. Inferencial

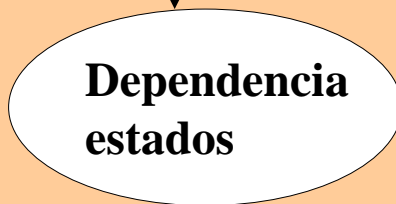


**Visible car
state**

concepto

Conoc. Dominio

*Mapeado
Inferencia-dominio*



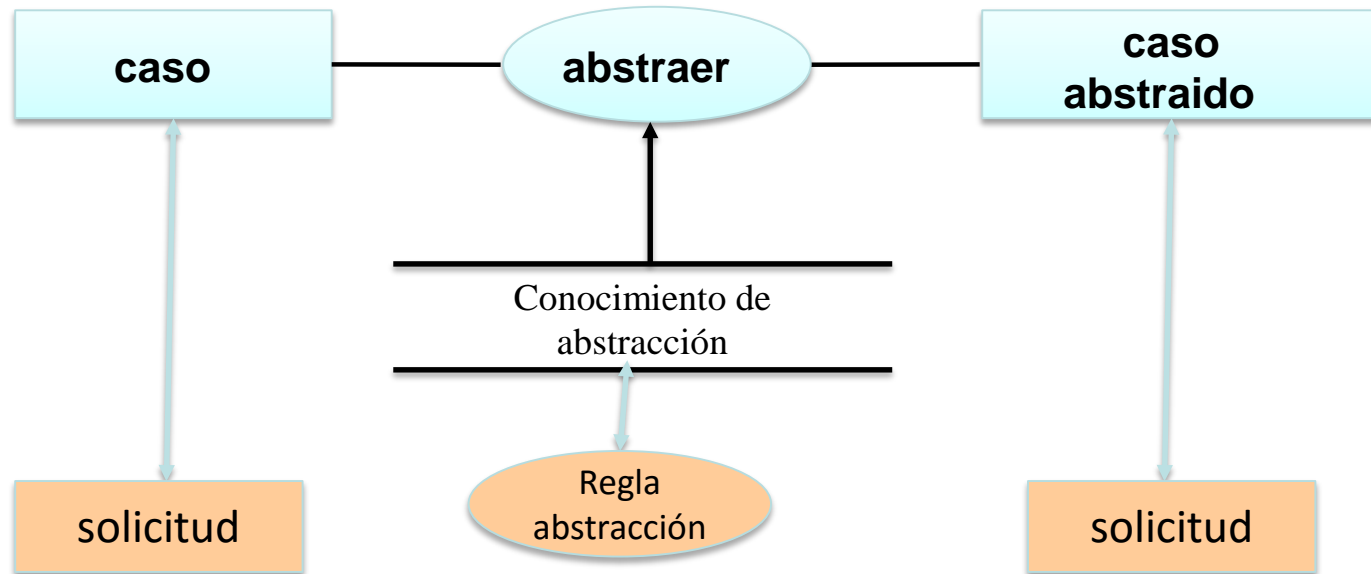
**Dependencia
estados**

Tipo de regla

**Invisible car
state**

concepto

Ejemplo asesoramiento ordenador: Aplicación inferencias-dominio



KNOWLEDGE-ROLE: conocimiento-abstracción;
 TYPE: STATIC;
 DOMAIN-MAPPING: regla-abstracción FROM asesoramiento-ordenador;
 END KNOWLEDGE-ROLE conocimiento-abstracción;

KNOWLEDGE-ROLE: caso;
 TYPE: DYNAMIC;
 DOMAIN-MAPPING: solicitud;
 END KNOWLEDGE-ROLE: caso;

KNOWLEDGE-ROLE: caso-abstraído;
 TYPE: DYNAMIC;
 DOMAIN-MAPPING: solicitud;
 END KNOWLEDGE-ROLE: caso-abstraído;



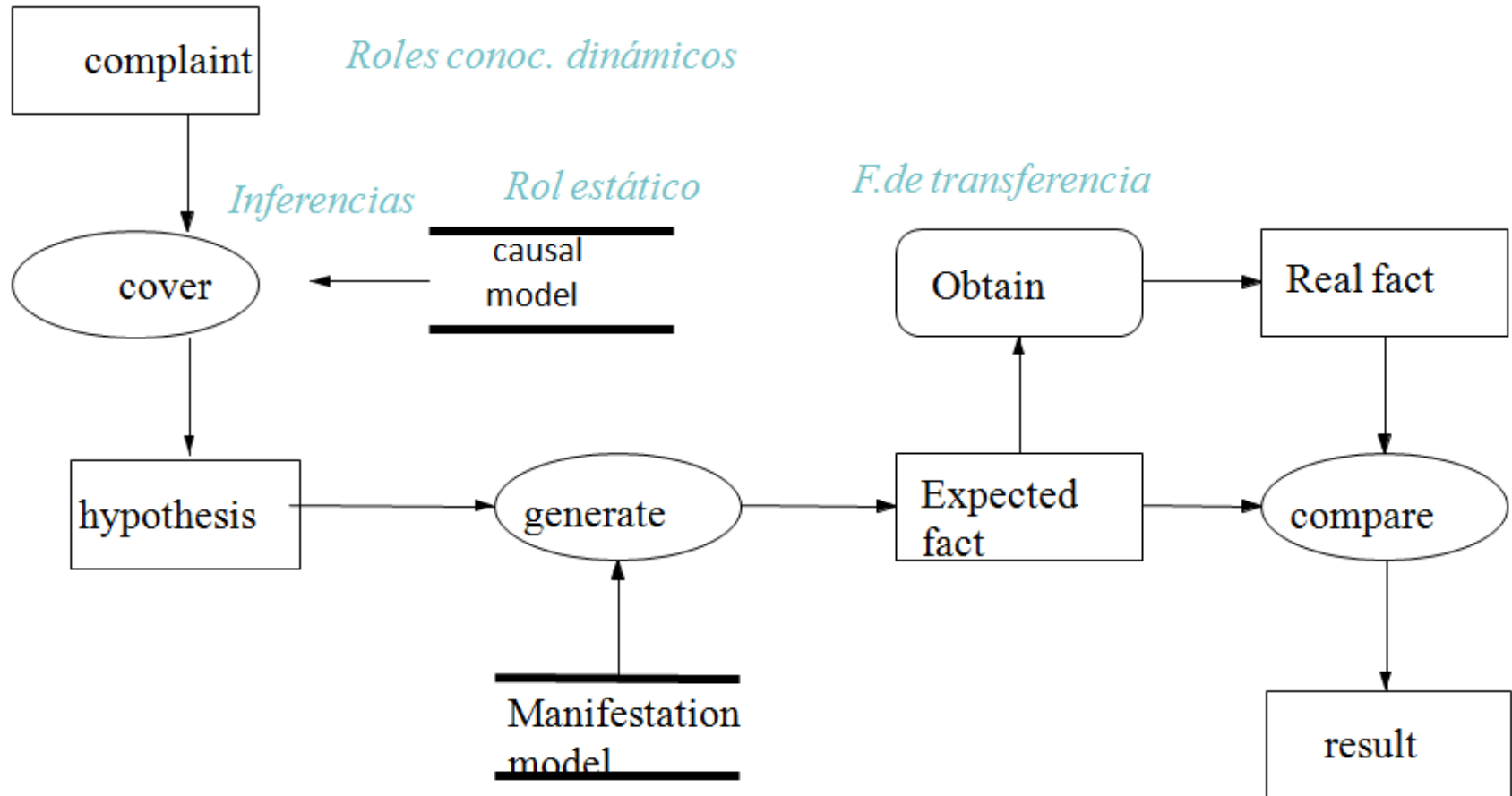
- Ventajas de la utilización de roles en la especificación
 - Desacoplar la especificación del Conocimiento del Dominio del de Inferencias
 - Posibilita llevar ambas especificaciones en paralelo
 - Vocabulario sobre cómo se utiliza el conocimiento independientemente del dominio de aplicación
 - **Reutilización**

- Codifican la transferencia de un item de información entre
 - el agente de razonamiento del modelo de conocimiento y
 - otro agente del mundo externo (usuario, otro sistema,...)
- desde el punto de vista del modelo de conocimiento es un caja negra:
 - sólo interesa su nombre y su E/S
- la especificación detallada de las funciones de transferencia es parte de otro modelo,
 - Modelo de comunicación
- Nombres estándar: **OBTAIN**, **RECEIVE**, **PRESENT**, y **PROVIDE**.

	Iniciativa sistema	Iniciativa externa
Información externa	OBTAIN	RECEIVE
Información interna	PRESENT	PROVIDE

```

TRANSFER-FUNCTION obtener;
    TYPE: OBTAIN;
    ROLE:
        INPUT: observable;
        OUTPUT: hallazgo;
END TRANSFER-FUNCTION obtener;
    
```

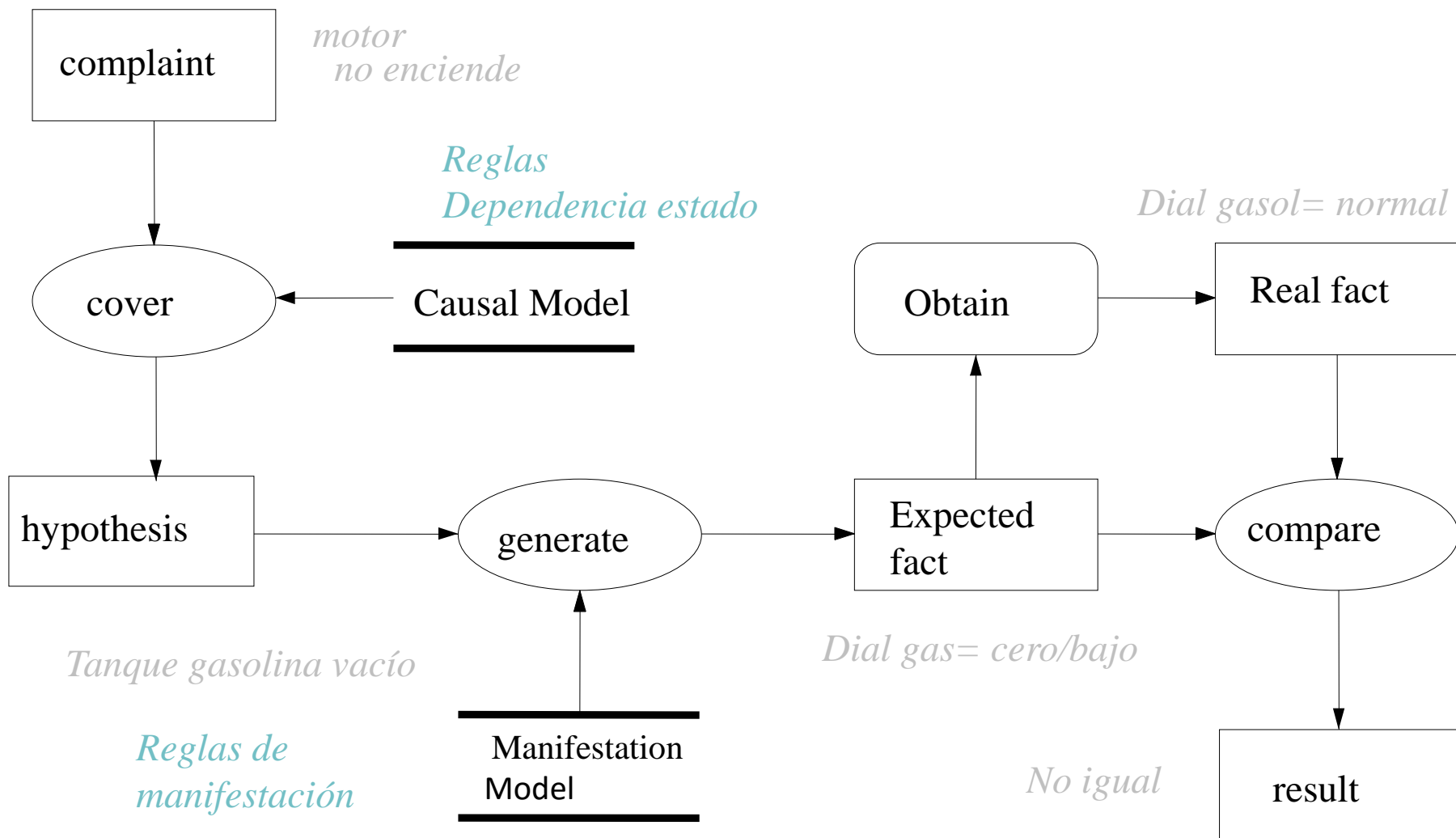




Uso de las Estructuras Inferenciales

- Representación abstracta de los pasos del proceso de razonamiento
- Vehículo importante de comunicación en el grupo durante el proceso de desarrollo
- A menudo son provisionales
- Suele ser útil realizar anotaciones con ejemplos del dominio.
- Define:
 - capacidades inferenciales del sistema,
 - el vocabulario y
 - las dependencias para el control
 - del control se ocupa el **conocimiento de la tarea**

Estructura Inferencial con anotaciones





Conocimiento tareas e inferencias en CML

TASK-KNOWLEDGE identificador

Descripción de tareas y métodos de tareas

END TASK-KNOWLEDGE

INFERENCE-KNOWLEDGE identificador;

Especificación de inferencias;

Especificación de roles de conocimiento;

Especificación de funciones de transferencia;

END INFERENCE-KNOWLEDGE identificador;



UNIVERSIDADE DA CORUÑA

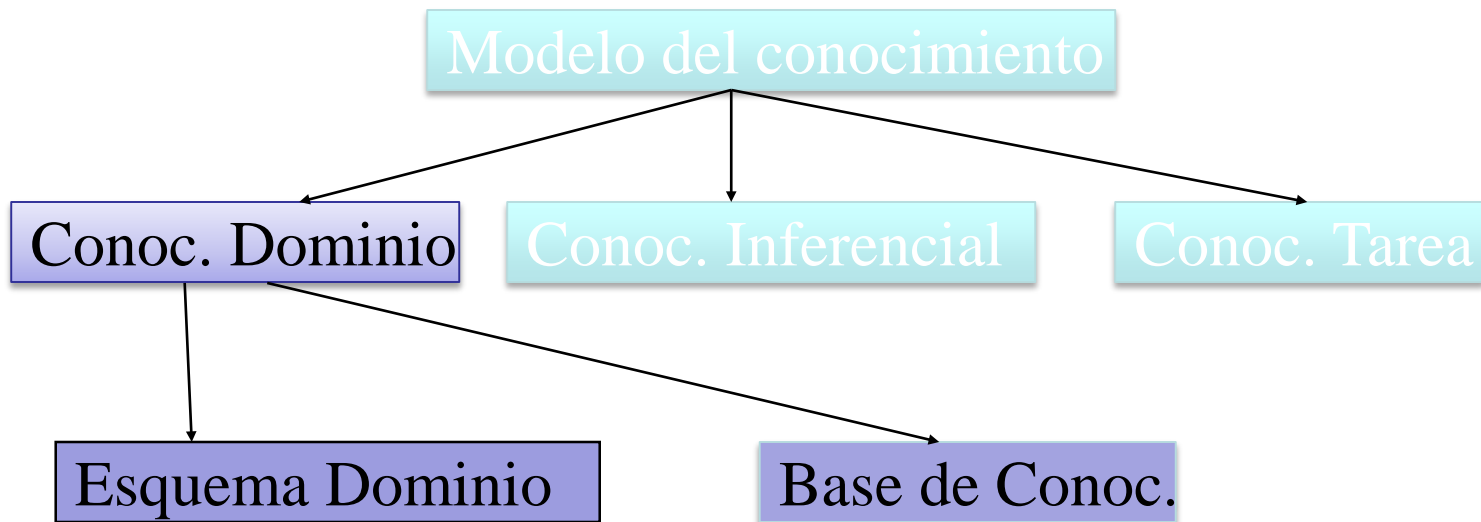


EL MODELO DEL CONOCIMIENTO.

Conocimiento del Dominio

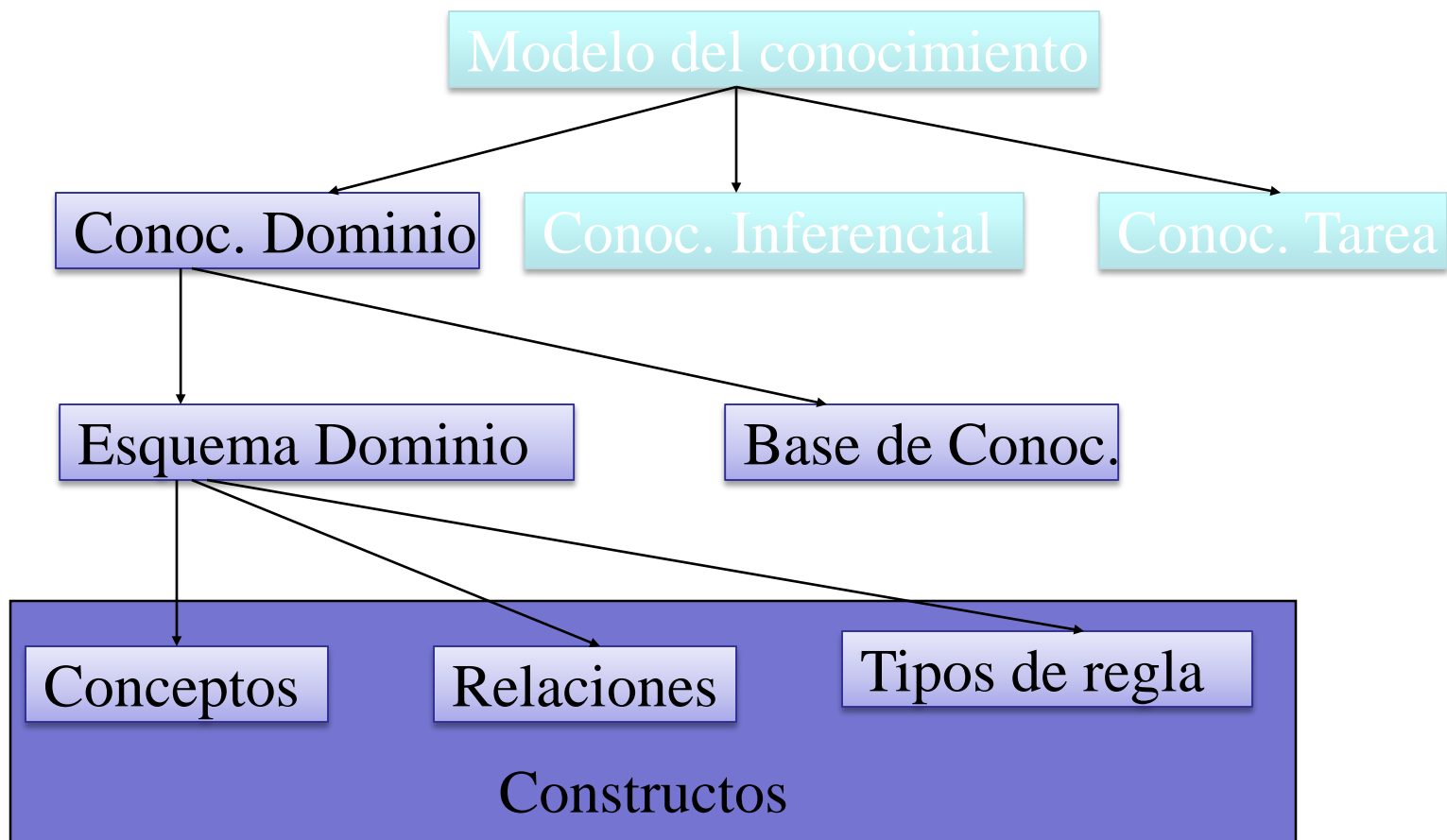
Desarrollo de Sistemas Inteligentes.

Modelo del conocimiento: Conocimiento Dominio





- Información estática y objetos de conocimiento en un dominio.
- Tiene dos partes:
 - Esquema del dominio
 - Describe la estructura estática de la información/conocimiento a través de definiciones tipo
 - comparable al modelo de datos/modelo de objetos en IS
 - definido a través de los constructos del dominio
 - Base de conocimiento
 - contiene instancias de los tipos que se especifican en el esquema del dominio (es decir, es un conjunto de instancias de conocimiento)
 - comparable al contenido de una bbdd





Constructos para el Esquema del Dominio

■ Conceptos:

- describen un conjunto de objetos o instancias del dominio que comparten características similares.
- como clases de objetos , pero sin operaciones ni métodos de O-O

■ Atributos

- valores primitivos. Características de los conceptos
- Un atributo puede tener un **Valor (value)**
- Los valores son atómicos y se definen a través de un **Tipo de Valor (value type)** , que establece los valores permitidos para el atributo (booleano, real, etc).
- El value type universal permite cualquier valor
- Por defecto, **cardinality** es 0-1

■ Relaciones

- Algunas Predefinidas: SUPERTIPO/SUBTIPO-DE y AGREGADO/PARTE
- como las Asociaciones en O-O

■ Tipos de Regla

- introduce expresiones => no hay equivalente en IS

Ejemplo: Conceptos de COCHE

Dial-gasolina

Value: valor-dial

```
CONCEPT Dial -gasolina;
  ATRIBUTTES:
    value: valor-dial;
END CONCEPT dial- gasolina;
```

```
VALUE-TYPE valor-dial;
  VALUE-LIST {cero, bajo, normal};
  TYPE: ORDINAL;
END VALUE_TYPE valor-dial;
```

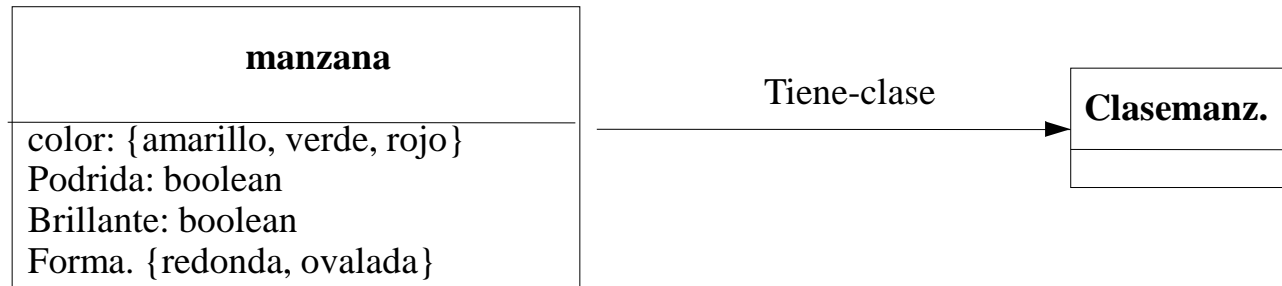
Tanque-gasolina

status: {lleno,
casi-vacío,
vacío}

```
CONCEPT tanque -gasolina;
  ATRIBUTTES
    status: {lleno, casi-vacío,vacío};
END CONCEPT tanque -gasolina;
```

```
CONCEPT cliente;
  DESCRIPTION:
    “ Un posible cliente”;
  ATTRIBUTES:
    nombre: STRING;
    domicilio: STRING;
    edad: NATURAL;
    ...
  AXIOMS:
    edad>=18;
END CONCEPT cliente;
```

Ejemplo: Concepto MANZANA



Los conceptos son el punto de comienzo para el modelado del conocimiento

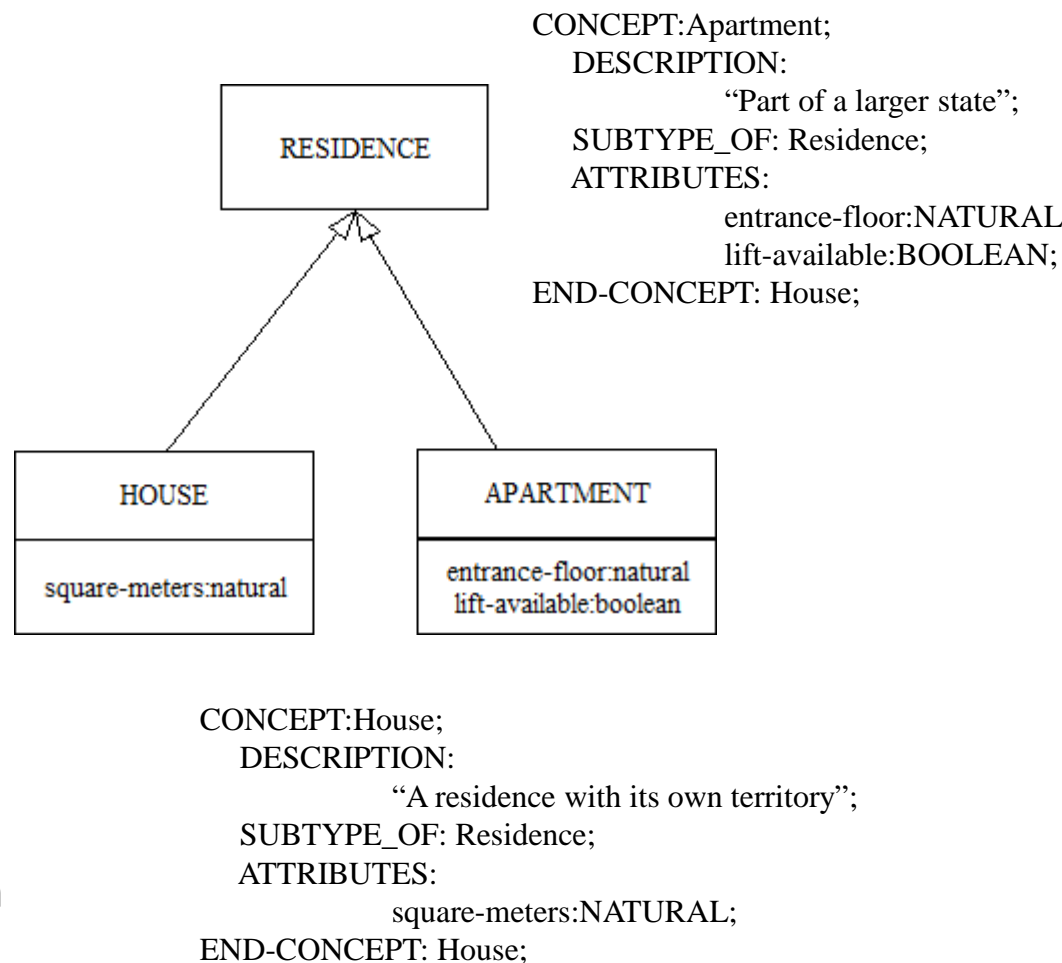
Granny Smith:
Clase manzana

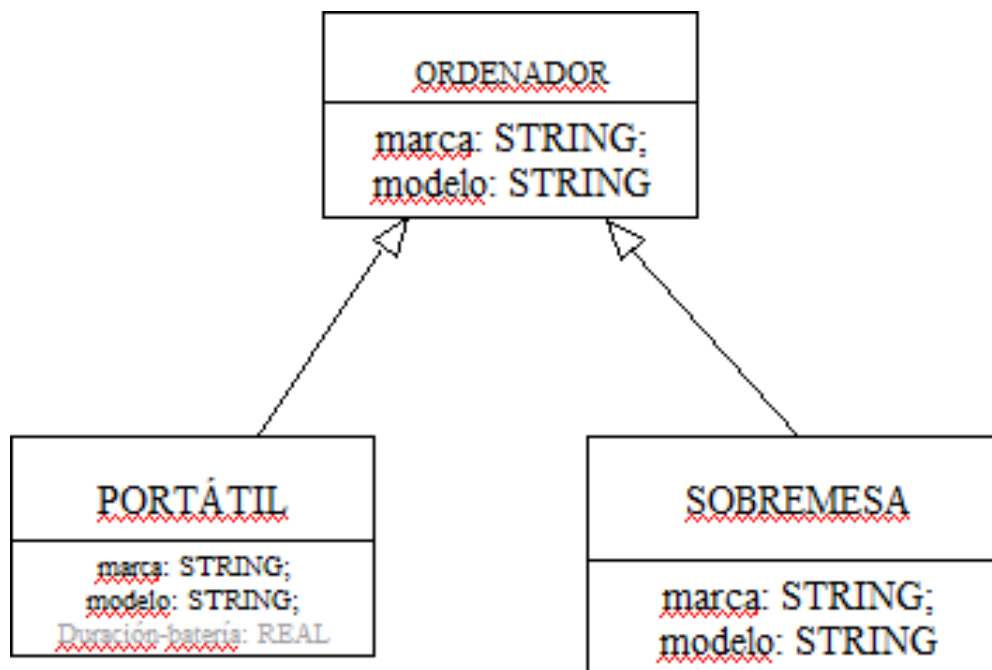
Golden Delicious:
Clase manzana

Grey Reinet:
Clase manz.

Present of England:
Clase manzana

- Los conceptos se pueden agrupar en jerarquías mediante el constructo **SUBTYPE-OF**.
 - Se coloca en la definición del subconcepto.
 - Los subconceptos heredan los atributos y relaciones del supertipo.
 - Pueden tener más atributos y relaciones propios.
- La definición de subtipos no está limitada a los conceptos, también valdría para las relaciones.
 - Ej. Una relación **propiedad** de un **vehículo** podría especializarse en **propiedad-coche** y **propiedad-bicicleta**.

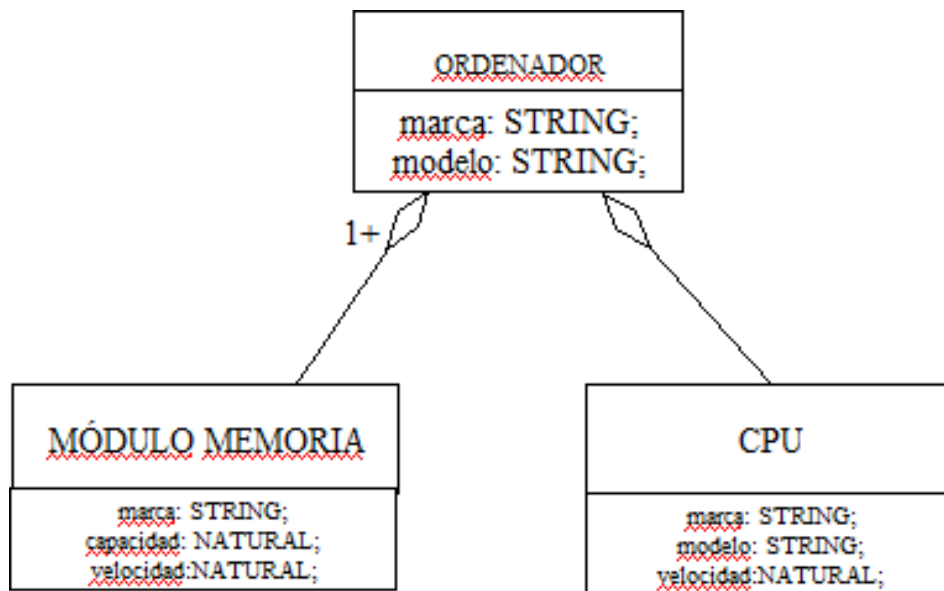




CONCEPT: ordenador;
DESCRIPTION:
 “ordenador personal”;
SUPERTYPE_OF: Portátil, Sobremesa;
SEMANTICS:
DISJOINT: YES;
COMPLETE: YES;
ATTRIBUTES:
 marca: STRING;
 modelo: STRING;
END-CONCEPT: Ordenador;

CONCEPT: portátil;
DESCRIPTION:
 “ordenador portátil”;
SUBTYPE_OF: Ordenador;
ATTRIBUTES:
 duración-batería: REAL;
END-CONCEPT: portátil;

CONCEPT: sobremesa;
DESCRIPTION:
 “ordenador no portátil”;
SUBTYPE_OF: Ordenador;
END-CONCEPT: sobremesa;



CONCEPT: ordenador;
DESCRIPTION:
 “ordenador personal”;
HAS-PARTS: CPU, módulo-memoria;
ATTRIBUTES:
 marca: STRING;
 modelo: STRING;
END-CONCEPT: ordenador;

CONCEPT: módulo-memoria;
DESCRIPTION:
 “Memoria RAM”;
PART-OF: Ordenador;
CARDINALITY: 1+;
ATTRIBUTES:
 marca: STRING;
 capacidad: NATURAL;
 velocidad :NATURAL;
END-CONCEPT: módulo-memoria;

CONCEPT: CPU;
DESCRIPTION:
 “Procesador”;
PART-OF: Ordenador;
CARDINALITY: 1+;
ATTRIBUTES:
 marca: STRING;
 modelo: STRING;
 velocidad :NATURAL;
END-CONCEPT: CPU;



- Pueden utilizarse en forma similar al diagrama entidad-relación
- Las relaciones entre conceptos se definen con el constructo **RELATION** o **BINARY RELATION**.
- Se definen a través de la especificación de **ARGUMENTOS (ARGUMENTS)**
- La **CARDINALIDAD (CARDINALITY)** se define para cada uno de los argumentos, y su valor de defecto es 1.
- También se puede especificar un **ROL (ROLE)** para cada argumento
- Las relaciones, al igual que los conceptos, también pueden tener **ATRIBUTOS (ATTRIBUTES)**

Especificación de relaciones binarias



```

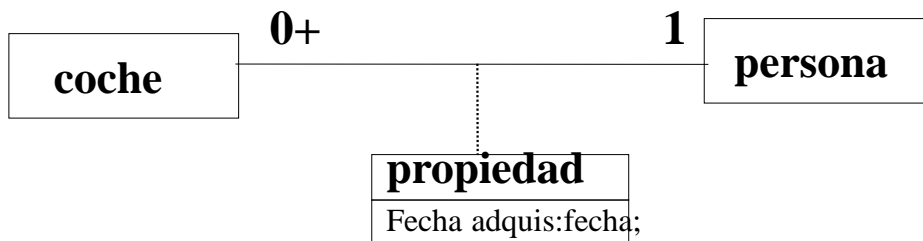
BINARY_RELATION perteneencia;
  ARGUMENT -1: coche;
  ARGUMENT-2: persona;
  CARDINALITY: ANY;
END BINARY_RELATION: perteneencia;
  
```



```

BINARY_RELATION propiedad-de;
  INVERSE: posee;
  ARGUMENT-1: coche;
  CARDINALITY: ANY;
  ARGUMENT-2: persona;
  CARDINALITY: 0-1;
END BINARY_RELATION propiedad-de;
  
```

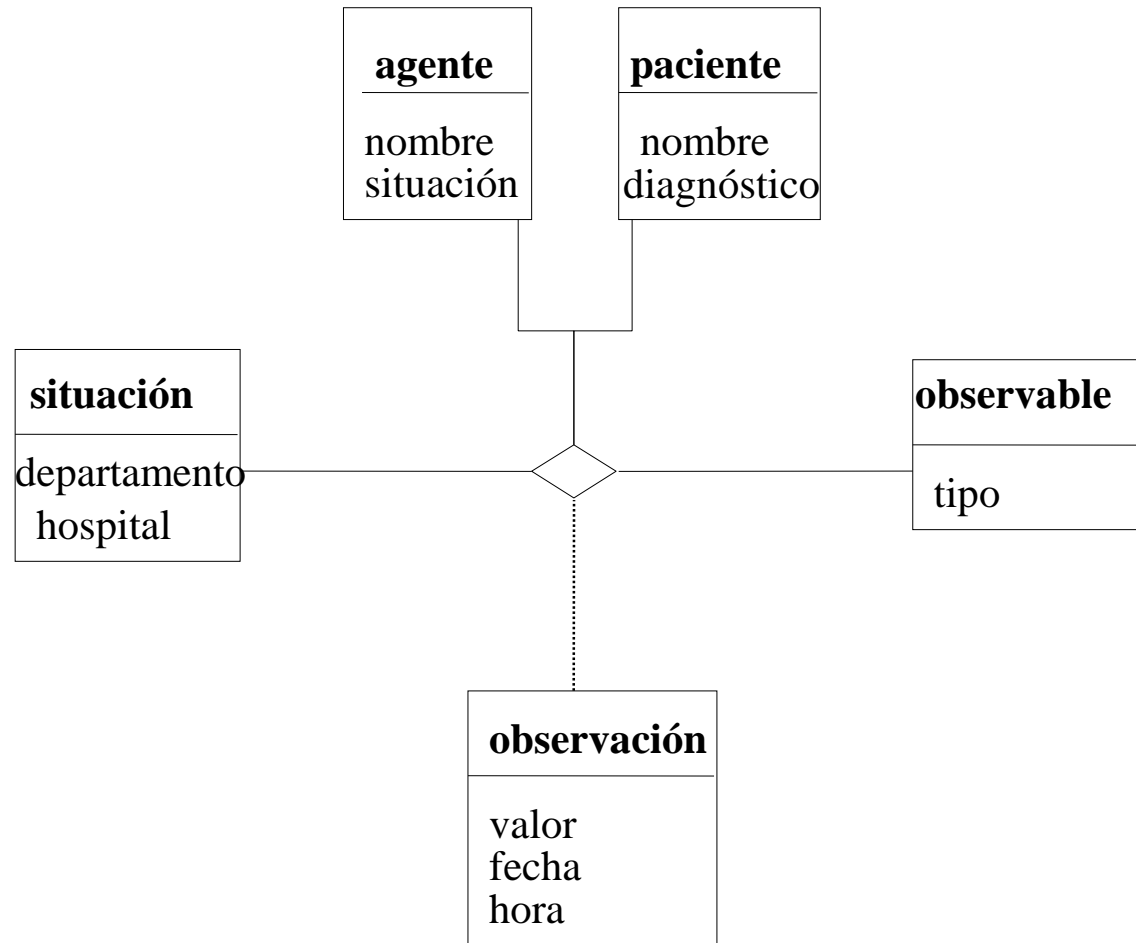
Materializada



```

BINARY-RELATION propiedad;
  ARGUMENT-1: coche;
  ARGUMENT-2: cliente;
  CARDINALITY: ANY;
  ATTRIBUTES:
    fecha-adquisición: tfecha;
END BINARY-RELATION: propiedad;
  
```

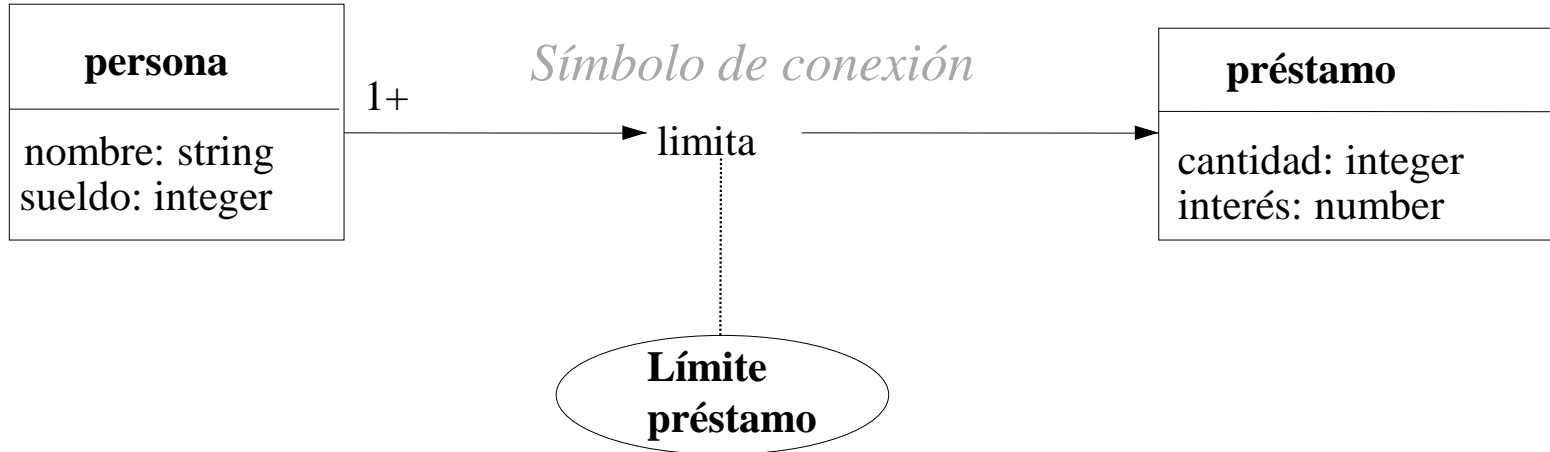
Relaciones de orden n





- Las reglas son una forma natural y común de conocimiento simbólico.
- Es una de las diferencias más importantes con los Modelos de Datos de la IS.
- Permiten estructurar el conocimiento adquirido del experto.
- Para modelar la estructura de las reglas se utiliza el constructo **Tipo de Regla**.
 - Son un tipo especial de relación. Representan dependencias entre los conceptos
 - Antecedente y Consecuente no son instancias de conceptos, sino expresiones de esas instancias.
- Se modela una relación entre expresiones acerca de los valores de los atributos.
 - dial-gasolina.valor= cero -> tanque -gasolina.estado = vacío
 - Tanque.gasolina=vacío-->Gasolina.motor=false
- Se modelan un conjunto de reglas de estructura similar
- El análisis del conocimiento se centra en encontrar reglas con una estructura común.
- Las relaciones no son estrictamente lógicas, es necesario especificar un **Simbolo de Conexión** entre antecedente y consecuente.

Ejemplo de especificación de un tipo de regla



<antecedente> <símbolo-conexión> <consecuente>

Persona.sueldo <= 10,000

LIMITA

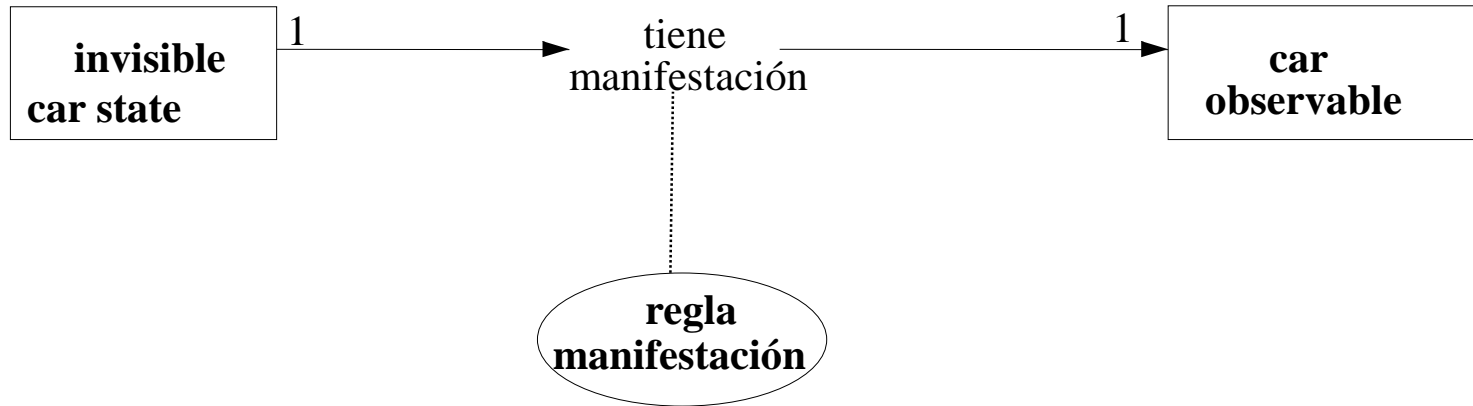
Préstamo.cantidad <= 2,000

Persona.sueldo > 10,000 AND persona.sueldo <= 20,000

LIMITA

Préstamo.cantidad <= 3,000

Diagnóstico coche: Tipos de reglas



RULE_TYPE regla-manifestación;

DESCRIPTION: “Regla que establece una relación entre un estado interno y un comportamiento externo en términos de un valor observable”;

ANTECEDENT:

invisible-car-state;

CONSEQUENT:

Car-observable;

CONNECTION-SYMBOL:

tiene-manifestación;

END RULE-TYPE regla-manifestación;



!!! MUY IMPORTANTE !!!

- El concepto de regla que utilizamos aquí no implica la utilización de este formalismo en la implementación
- La representación final de este conocimiento en nuestro sistema software se decidirá más tarde durante la fase de diseño
- Las reglas en este nivel sólo son un vehículo de análisis para capturar dependencias lógicas que puedan ocurrir en el dominio



Esquema del dominio. Especificación en CML

■ Estructuración

```
DOMAIN-SCHEMA asesoramiento-ordenador;
    Especificación de conceptos;
    Especificación de relaciones;
    Especificación de atributos;
    Especificación de tipos de reglas;
END DOMAIN-SCHEMA asesoramiento_ordenador;
```

■ Reutilización

- Importar todas las construcciones definidas en otro(s) esquema(s):

```
DOMAIN-SCHEMA asesoramiento-ordenador;
USES: componentes-del-ordenador; /* esquema importado */
...
END DOMAIN-SCHEMA asesoramiento-ordenador;
```

- Importar sólo algunas construcciones definidas en otro(s) esquema(s):

```
DOMAIN-SCHEMA asesoramiento-ordenador;
USES: módulo-memoria, CPU FROM componentes-del-ordenador;
    categoría-edad FROM características-cliente;
```


- El esquema del dominio describe tipos de conocimiento del dominio :
 - conceptos
 - relaciones
 - Tipos de reglas
- Una **base de conocimiento** contiene instancias de los tipos de conocimiento definidos en el esquema del dominio
 - Las instancias de los tipos-de-reglas contienen reglas.
- Especificación en dos partes:
 - **slot USES:**
 - <tipos usados> de <esquema>
 - **slot EXPRESSIONS:**
 - <instancias>
 - representación instancias:
 - semiformalmente, símbolo de conexión separando antecedente y consecuente
 - formalmente

KNOWLEDGE-BASE red-coche;

USES:

dependencia-estado FROM esquema-diagnóstico-coche;

manifestación-regla FROM esquema-diagnóstico-coche;

EXPRESSIONS::

/* dependencias estado*/

fusible. estado = quemado CAUSA motor.estado= off;

batería.estado = baja CAUSA motor.estado= off;

....

/* manifestation reglas */

fusible.estado= quemado TIENE-MANIFESTACIÓN

fusible-inspección.valor = roto;

batería.estado = bajo TIENE-MANIFESTACIÓN

batería-dial.valor = cero;

END KNOWLEDGE-BASE red-coche;

- Separación entre esquema del dominio y las bases de conocimientos
- Adquisición de Conocimiento:
 - Definición de los *tipos* de conocimiento, como los tipos de reglas.
 - Definición de las ocurrencias, y colocarlas en la base de conocimiento.

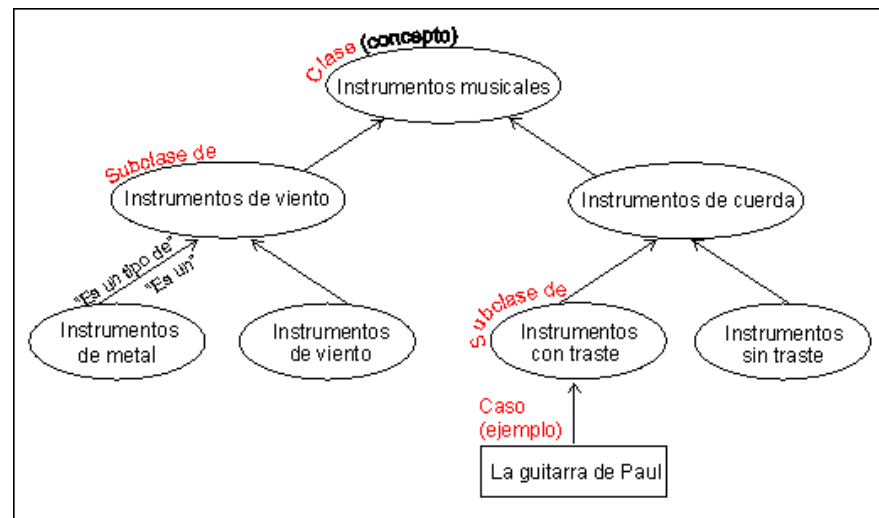
DOMAIN-KNOWLEDGE identificador;

DOMAIN-SCHEMA identificador;
Especificación de conceptos,
atributos, relaciones y tipos de reglas;
END DOMAIN-SCHEMA identificador;

KNOWLEDGE-BASE identificador;
Ocurrencias de objetos del esquema
END KNOWLEDGE-BASE identificador;

END DOMAIN-KNOWLEDGE identificador;

- Especificación explícita y formal de una conceptualización consensuada
- Descripción formal de conceptos en un dominio.
- El desarrollo de una ontología incluye:
 - Definir clases en la ontología
 - Situar las clases en un jerarquía de taxonomías (subclase-superclase)
 - Definir [slots –atributos–] y describir los valores permitidos para esos [slots]
 - Rellenar los valores de los slots con ejemplos.



Identificación conocimiento

- Familiarización con el dominio
(fuentes de información, glosario, escenarios típicos)
- Lista potencial de los componentes del modelo reutilizables
(componentes relacionados con la tarea y con el dominio)



Especificación conocimiento

- Escoger plantilla de tarea
(proporciona la descomposición inicial de la tarea)
- Construir la conceptualización inicial del dominio
(principales tipos de información del dominio)
- Especificación completa del modelo del dominio
(modelo del conocimiento con BC parciales)



Refinado conocimiento

- Validar modelo conocimiento
(simulación en papel, prototipo de razonamiento del sistema)
- Refinado de la base de conocimiento
(completar las bases de conocimiento)



■ METAS:

- Validar el modelo del conocimiento
- “Rellenar” las bases de conocimiento

■ ACTIVIDADES:

- Actividad 3.1.- Completar las bases de conocimiento.
- Actividad 3.2.- Evaluar el modelo de conocimiento.



Actividad 3.1.-Completar las bases de conocimiento

- Completar también actúa como un test de validación del esquema del dominio.
- Usualmente no es posible definir bases de conocimiento completas y correctas en el primer ciclo.
- Las bases de conocimiento necesitan mantenimiento
 - el conocimiento cambia con el tiempo
- Técnicas:
 - incorporar herramientas de edición para actualización de BCs, usar transcripts, trazados, entrevistas estructuradas, aprendizaje automático, etc.



Actividad 3.2.-Validar el modelo de conocimiento

■ Interna

- verificación = evaluación interna
 - “¿está correcto el modelo?”
- Técnicas
 - recorridos estructurados (walk-throuhgs)
 - herramientas software que comprueban sintaxis

■ Externa

- validación = evaluación frente a los requisitos del usuario
 - “¿es el modelo correcto?”
- mucho más difícil y compleja
- Técnica principal: simulación
 - simulación en papel
 - sistema prototipo

Validación del MC: Simulación en papel

DOMINIO (escenario)	MODELO (mapeado)	EXPLICACIÓN
the user says: "the car does not start"	DIAGNOSIS: <u>Complaint</u> : engine- behavior.status = does-not-start	Complaint is received, for which a diagnostic task is started
a possible cause is that the fuel tank is empty	COVER: <u>hypothesis</u> ; fuel- tank.status = empty	One of the three possible causes is produced.
in that case we would expect the gas indicator to be low	PREDICT: <u>expected-finding</u> : gas-dial.value = zero	The expected finding provides us with a way of getting supporting evidence for hypothesis

Ejemplo de validación mediante prototipo

Assessment of a residence application

Tasks

Transaction "order_assessment" is active
Transaction "get_case" is active
Transaction "get_case" is finished
Task "assess_case" has been activated
Task "abstract_case" has been activated

Inferences

Inference "abstract" has started
Inference "abstract" has succeeded
Inference "abstract" has started
Inference "abstract" has succeeded

Blackboard

```

residence:num_rooms=4
residence:rent=442
residence:min_num_inhabitants=2
residence:max_num_inhabitants=4
residence:subsidy_type=subsidizable
residence:surface_in_square_meters=94
applicant:registration_number=xyza-240690
applicant:applicant_type=starter
applicant:name=N.N.
applicant:street_address=IJzerlaan 10
applicant:city=Utrecht
applicant:age=21
applicant:gross_yearly_income=36000
applicant:household_size=2
applicant:household_type=multi_person
New value for role "abstracted_case":
residence:number=11274
residence:category=starter_residence
residence:build_type=house
residence:street_address=Van Houtlein 5
residence:city=Utrecht
residence:num_rooms=4
residence:rent=442
residence:min_num_inhabitants=2
residence:max_num_inhabitants=4
residence:subsidy_type=subsidizable
residence:surface_in_square_meters=94
applicant:registration_number=xyza-240690
applicant:applicant_type=starter
applicant:name=N.N.
applicant:street_address=IJzerlaan 10
applicant:city=Utrecht
applicant:age=21
applicant:gross_yearly_income=36000
applicant:household_size=2
applicant:household_type=multi_person
applicant:age_category=upto 22
          
```

Continue



- Punto de vista CKads:
 - no es diferente del desarrollo
- El desarrollo del modelo es un proceso cíclico
- Los modelos actúan como repositorios de información
 - continuamente actualizados
- Consecuencia : mayores requisitos de las herramientas de soporte
 - herramientas de transformación



- Especificación del modelo de conocimiento
- Lista de todas las fuentes de información que se usan.
- Lista de los componentes del modelo que tenemos en cuenta para reutilización.
- Escenarios para la resolución del problema de la aplicación a desarrollar
- Resultados de las simulaciones realizadas durante la validación
- Material de elicitación (apéndices)


Identificación conocimiento

Familiarización con el dominio de aplicación



Especificación conocimiento

Análisis detallado del conocimiento
asistido por modelos reutilizables

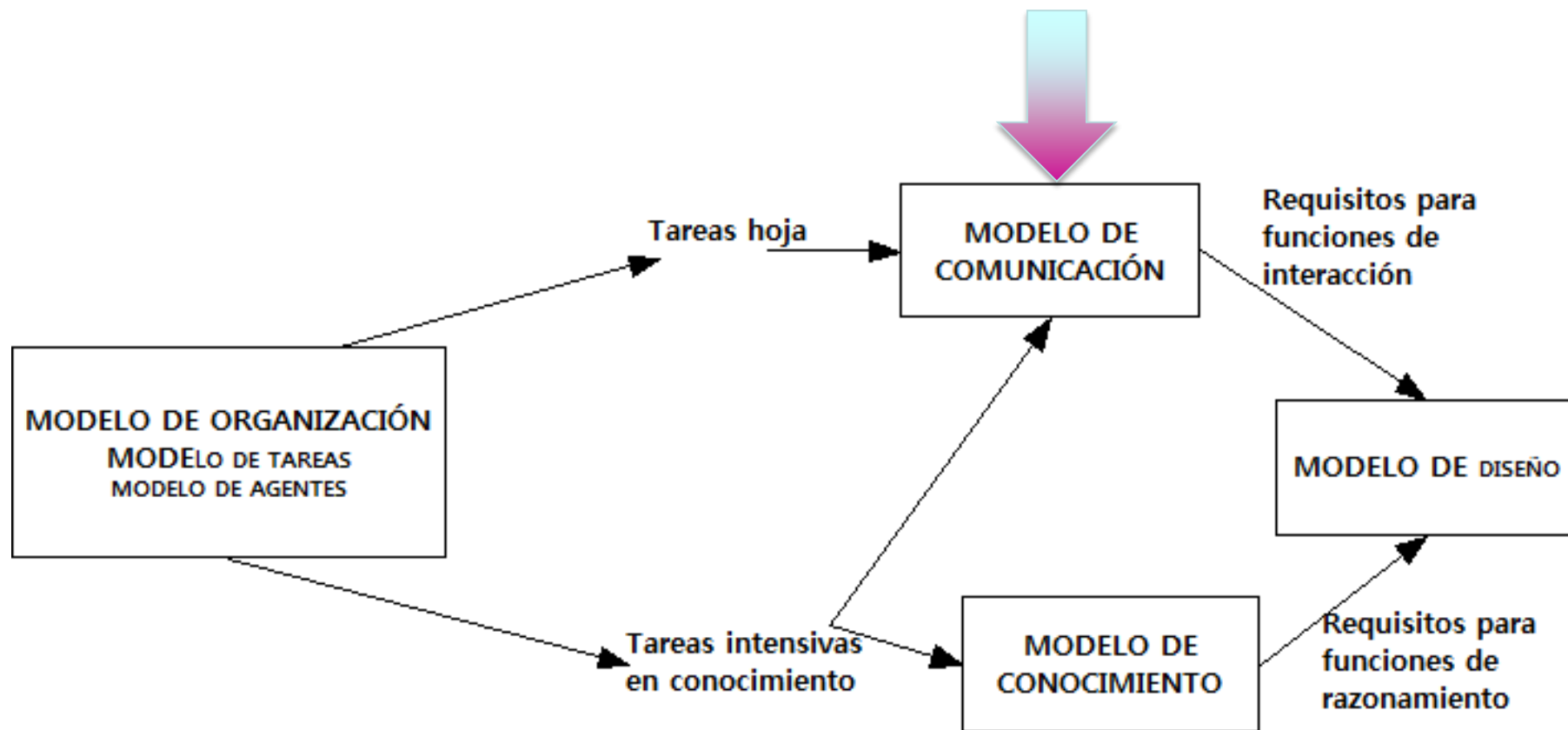


Refinado conocimiento

Completar y validar el modelo de conocimiento



Modelo de Comunicación

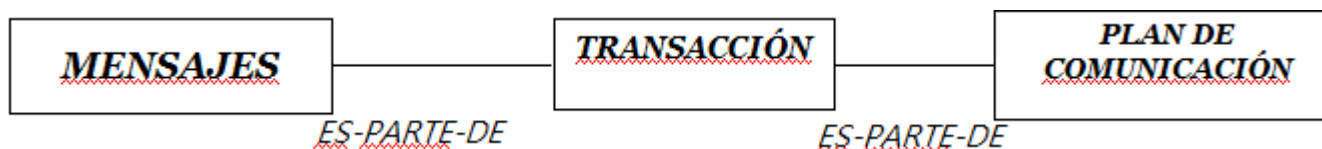




- Especifica procesos de interacción entre agentes para transferencia de información/conocimiento.
- Modelo de Comunicación = Especificación conceptual de:
 - ¿Qué información se transmite?
 - ¿Qué tipo de objetos de información se intercambian ?
 - ¿Entre qué agentes y tareas?
 - ¿Cómo?
- Control de nivel superior sobre la ejecución de la tarea.
 - múltiples tareas intensivas en conocimiento
- Tareas de comunicación adicionales
 - facilidades de explicación
 - Ejemplo: interacción básica sistema-usuario



- El Modelo de Comunicación especifica el intercambio de información entre las tareas llevadas a cabo por diferentes agentes.
- Tres capas consecutivas:
 1. **Plan general:**
 - Define el diálogo completo entre dos agentes
 2. **Transacciones individuales:**
 - Pueden consistir en uno o más mensajes que se detallan en el nivel 3 (más específico).
 - Existen patrones y tipos de comunicación predefinidos que permiten construir protocolos de mensajes de forma estructurada.
 3. **Especificación detallada de los intercambios de información**



■ Modelo de tareas

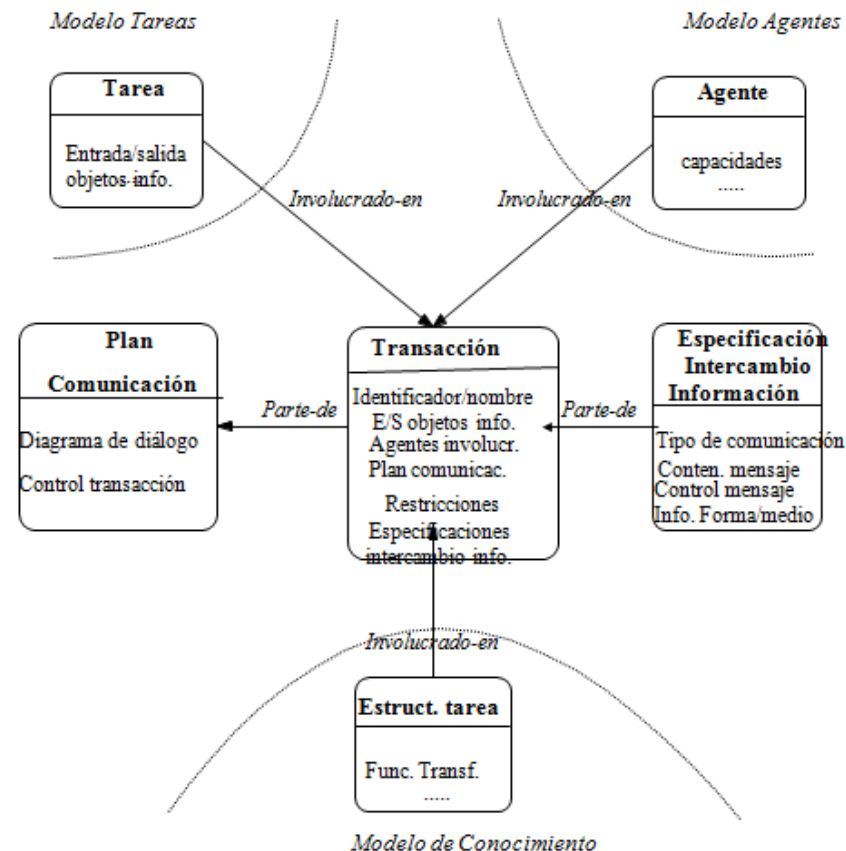
- lista de tareas hoja llevadas a cabo por los agentes considerados:
 - lista de tareas,
 - objetos de información de entrada y salida,
 - y su asignación a agentes específicos.

■ Modelo de conocimiento

- funciones de transferencia

■ Modelo de agentes

- descripción de los agentes relevantes: capacidades, responsabilidades, restricciones.





Plan de comunicación

- gobierna el diálogo completo entre los agentes
- organización de las transacciones

Transacción

- describe qué objetos de información se intercambian
- indica agentes y tareas implicadas
- Se intercambia entre dos tareas (hojas) llevadas a cabo por diferentes agentes
- es el bloque de construcción para el diálogo completo entre dos agentes.
- Las transacciones tienen una estructura interna
 - ejemplo: *obtain*

Especificación del intercambio de información

- detalla la estructura de la transacción
- consiste en mensajes
- sólo es necesaria para comunicaciones complejas

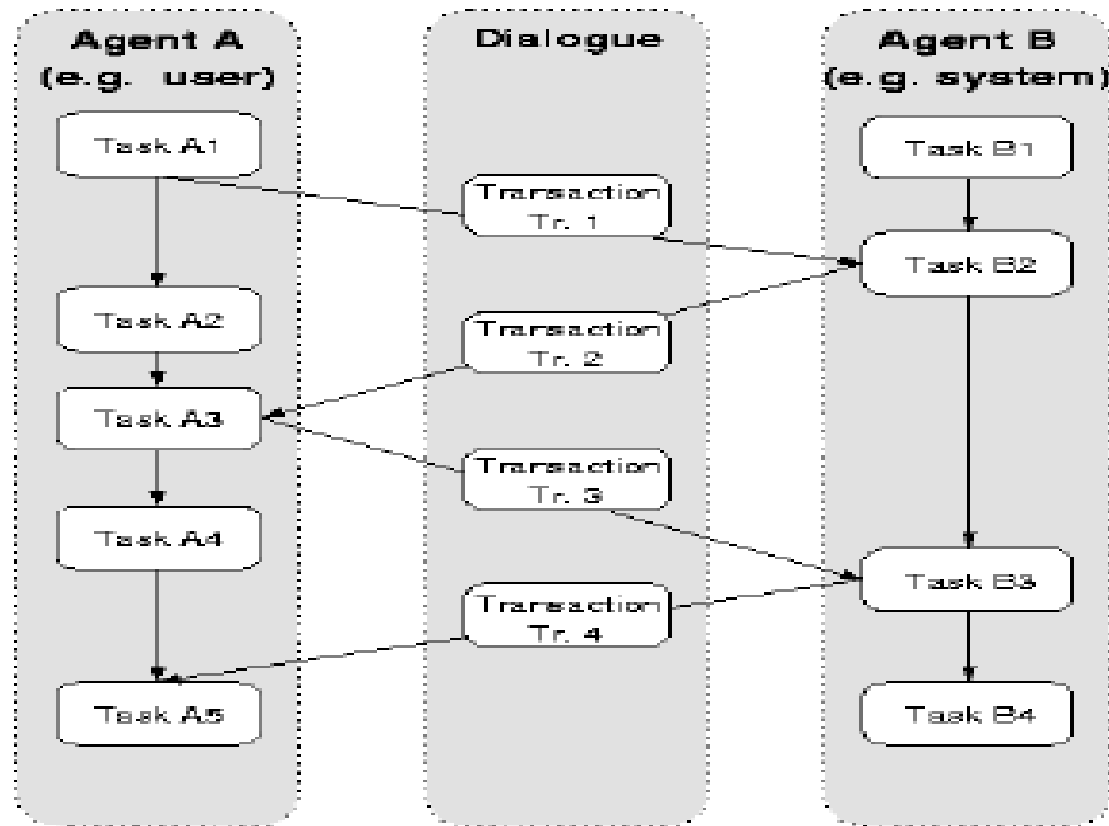


- **Descripción:** Diálogo de alto nivel que realizan dos agentes para llevar a cabo una tarea conjunta.
- Transacciones típicas
 - por ejemplo para un agente SBC y un usuario humano para una sesión del sistema:
 - entrada de datos
 - contestar o realizar preguntas
 - presentación de resultados de razonamiento
 - explicación de resultados
- Las **entradas del modelo** son:
 - **TM:** listado de las tareas hojas de cada agente junto con el elemento de información que intercambian
 - **KM:** conjunto de funciones de transferencia
 - **AM:** descripción de los agentes:
 - Capacidades y conocimiento
 - Responsabilidades y limitaciones



- **Para cada agente:**
 - Lista de todas las tareas hoja/ funciones de transferencia,
 - Seleccionar aquellas en las que intercambia información con otros agentes
- **Para cada tarea:**
 - Identificar conjunto de transacciones agente-agente asociadas
 - Identificar de la lista anterior el conjunto de transferencias agente-agente asociadas (sólo identificación). Nombrarlas
- El resultado se combina en un “**Diagrama de Diálogo**” (DD)
 - DD representa las transacciones entre dos agentes
- Se dibuja un DD para cada combinación de dos agentes que intercambien una cantidad de información razonable.
- Especificar el **control** sobre las transacciones
 - (pseudo código : primitivas de control SEND, RECEIVE, CARRY-OUT, etc; o
 - diagrama de transición de estados usando UML)

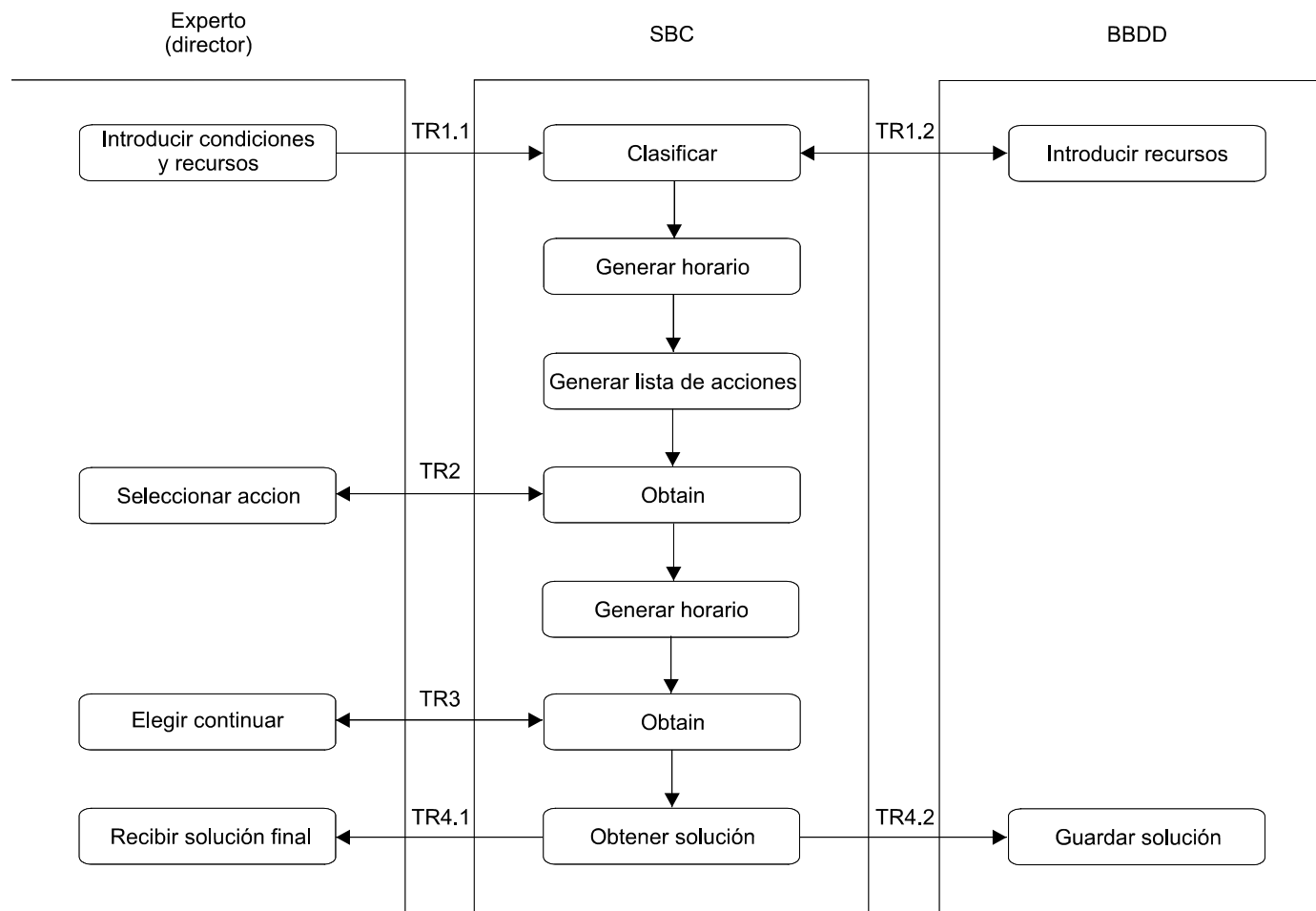
Diagrama de diálogo (DD): Estructura general



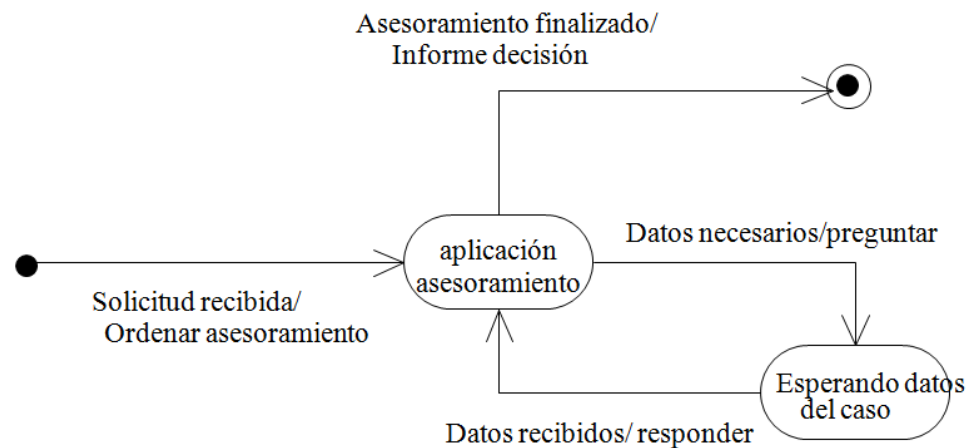
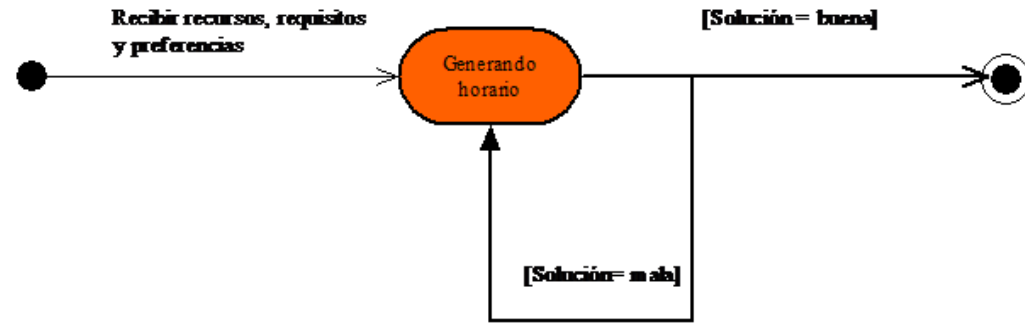
Nota: las tareas (hoja) de los agentes son clave para la construcción del DD

Ejemplo para Sistema de Horarios

Plan de comunicaciones.DD



- Diagramas de transición de estados (UML)
- Pseudo-código con primitivas de control especiales
 - SEND, RECEIVE
 - CARRY-OUT (combinación SEND/RECEIVE)
 - WAIT-until/while
 - PROCESS (tarea)
 - ; (secuencia)
 - REPEAT-until/while
 - IF THEN ELSE
 - & (AND), | (elección), V (OR)



Debe definirse una transacción para cada objeto de información que deba transmitirse a otro agente y que sea:

- una salida de una tarea hoja en TM ,
- una función de transferencia en KM



Formulario CM-1

MODELO DE COMUNICACIÓN	FORMULARIO CM-1
Nombre de la transacción	Nombre y breve explicación del objetivo de la transacción
Objetos de información	Objeto central de información, y entre qué tareas se transmite.
Agentes Involucrados	Agente emisor y agente receptor del objeto de información
Plan de Comunicaciones	Plan del que forma parte la transacción
Restricciones	Requisitos y precondiciones que deben darse para que pueda realizarse la transacción Postcondiciones que pasan a ser válidas después de que se produzca la transacción
Especificación del intercambio de información	Una transacción puede constar de varios mensajes de información de tipos diferentes, o manejar objetos de información adicionales de soporte, como explicaciones o ayudas. Se detallan en el formulario CM-2 correspondiente.



- Especificación detallada del mensaje:
 - 1. Contenido (locución): mediante una ***declaración proposicional***
 - 2. Intención (ilocución): mediante un mensaje escrito
- Tipos predefinidos (propósito):
 - **Delegación tareas**: Solicitar (*Request*); Exigir (*Require*); Ordenar (*Order*); Rechazar-td (*Reject-td*). Reject task-delegation
 - **Adopción tareas**: Proponer (*Propose*); Ofrecer (*Offer*); Acordar (*Agree*); Rechazar-ta (*Reject-ta*) Reject task adoption
 - **Intercambio de información puro**: Preguntar (*Ask*); Responder (*Reply*); Informar (*Inform*); Enviar-informe (*Report*)
- Nota: Intención = propósito x cometido
 - Cf. performatives in KQML (DARPA Knowledge Sharing Effort) and esp. COSY (Daimler-Chrysler)



PROPÓSITO: Delegación/adopción de tareas

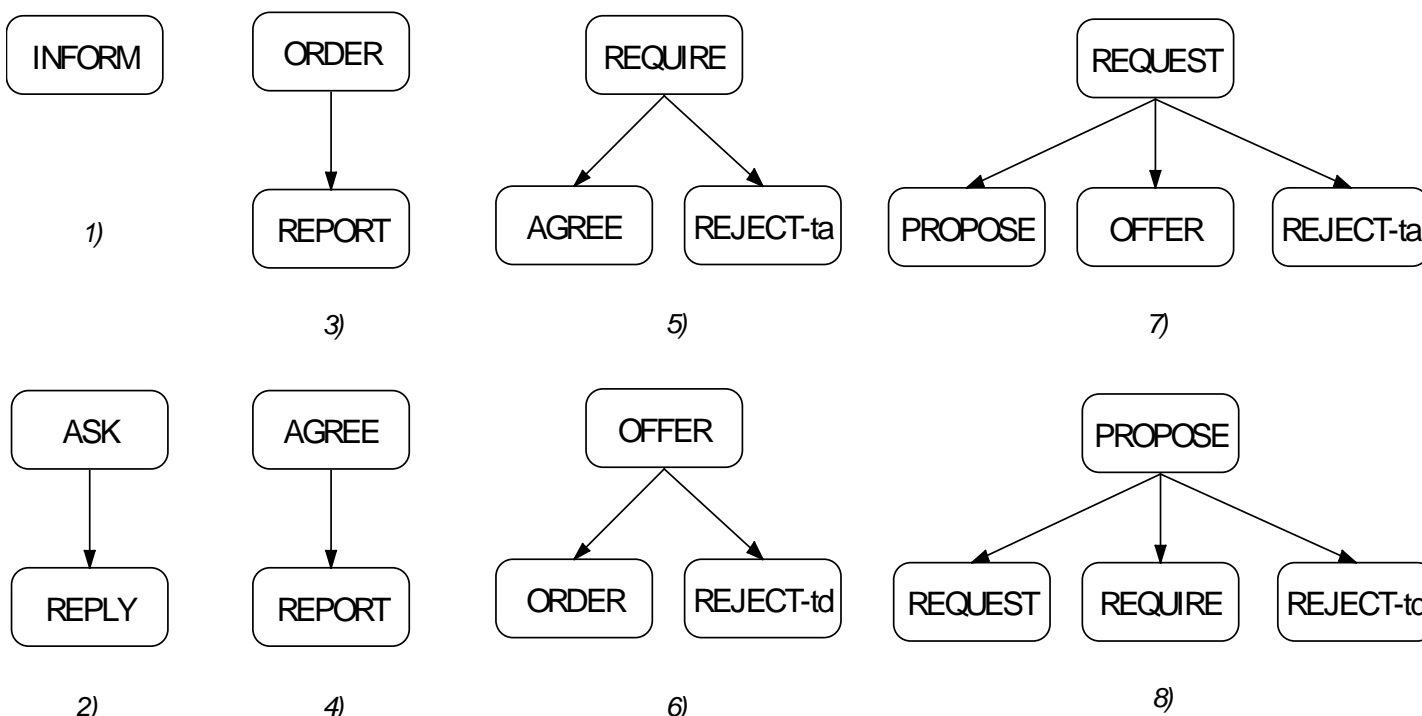
- **Request/Propose**: *Un agente ve en otro un potencial para la cooperación, pero el agente quiere negociar los términos. De forma coloquial “Estoy interesada, pero no quiero comprometerme todavía”. Existe interés, no compromiso.*
- **Require/Offer**: *el agente emisor ya ha realizado un pre-compromiso, y tiene la intención de preguntar acerca de su compromiso al agente receptor. Este tipo denota un compromiso condicional.*
- **Order/Agree**: *el agente ha realizado un compromiso, y actuará de acuerdo con ello.*
- **Reject-ta**: *denota que el agente no quiere comprometerse o cooperar en la delegación o adopción de tareas.*

▪ PROPÓSITO: Transmisión de Información

- **Ask/Reply**: *tiene como intención realizar una pregunta (query) sobre información de otro agente, y devolver a cambio otra información.*
- **Report**: *escribe un mensaje que envía después de que un agente haya actuado hacia la realización de una tarea meta en la que previamente había acuerdo, con la intención de dejar que el otro agente conozca el estado logrado (éxito, fallo, salida de la acción)*
- **Inform**: *se refiere a un tipo de mensaje que simplemente envía objetos de información a otro agente. Indica una acción informativa independiente: no hay involucradas peticiones o acuerdos previos (agree, request)*

- No sólo es posible escribir mensajes únicos
- Se pueden formar cadenas naturales de tipos de mensajes (cf. COSY):
Conversation policies

Patrones Comunicación



Ejemplos de Transacción

Nombre transacción	Ordenar aplicac. asesoramiento
Objetos Información	Una petición de residencia
Agentes involucrados	Entrada datos + sistema conocimiento (+ asignador)
Plan Comunicaciones	Plan asignación viviendas
Restricciones	Activo cuando llega la petición Prototipado: interactuar con el usuario
Intercambio Inform.	La transacción es del tipo "order"

Nombre transacción	Obtener datos aplicación
Objeto Información	Pares atributo-valor de bases de datos de solicitantes y residencias + sistema conocimiento
Agentes involucrados	Bases de datos+SBC
Plan Comunicaciones	Transacciones conectadas al estado "esperando datos" Plan asignación viviendas
Restricciones	Asegurar el mapeo de requisitos de datos en el formato de datos de la BBDD.
Intercambio Informac.	Transacción es del tipo "Ask-Reply"

Nombre Transacción	TR1.1.-Introducir requisitos de usuario
Objetos de Información	Los requisitos (restricciones y preferencias) y los recursos
Agentes Involucrados	Emisor: Director Receptor: SBC
Plan de Comunicaciones	Horarios (figuras anteriores)
Restricciones	Para iniciar la transacción, el SBC debe estar listo para ser usado.
Especificación del Intercambio	Es del tipo ORDER.

Nombre Transacción	TR1.2.- Obtener recursos
Objetos de Información	Los requisitos (restricciones y preferencias) y los recursos
Agentes Involucrados	Emisor: SBC Receptor: BBDD
Plan de Comunicaciones	Horarios
Restricciones	SBC listo y datos disponibles en la BBDD.
Especificación del Intercambio	Es del tipo REQUIRE.

Nombre Transacción	TR2.- Seleccionar acción
Objetos de Información	Lista de posibles acciones
Agentes Involucrados	Emisor: SBC Receptor: Director (usuario)
Plan de Comunicaciones	Horarios
Restricciones	El horario no está terminado y los requisitos están clasificados correctamente.
Especificación del Intercambio	Tipo ASK/REPLY

Nombre Transacción	TR3.- Confirmar solución
Objetos de Información	Horario final
Agentes Involucrados	Emisor: SBC Receptor: Director (usuario)
Plan de Comunicaciones	Horarios
Restricciones	Se inicia la transacción cuando el SBC tiene una solución.
Especificación del Intercambio	Tipo ASK/REPLY

Nombre Transacción	TR4.1.- Obtener solución
Objetos de Información	Horario final
Agentes Involucrados	Emisor: SBC Receptor: Director (usuario)
Plan de Comunicaciones	Horarios
Restricciones	Se inicia la transacción cuando el SBC tiene una solución satisfactoria.
Especificación del Intercambio	Tipo REPORT.

Nombre Transacción	Tr 4.2.- Guardar solución
Objetos de Información	Horario final
Agentes Involucrados	Emisor: SBC Receptor: BBDD
Plan de Comunicaciones	Horarios
Restricciones	Se inicia la transacción cuando el SBC tiene una solución satisfactoria.
Especificación del Intercambio	Tipo REPORT.

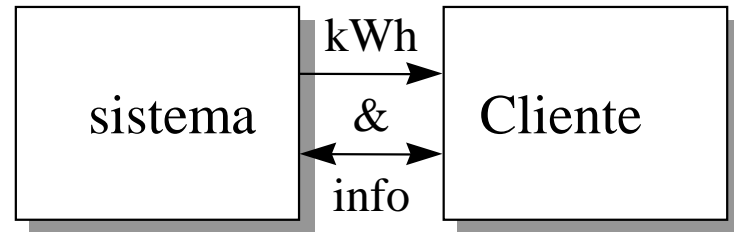
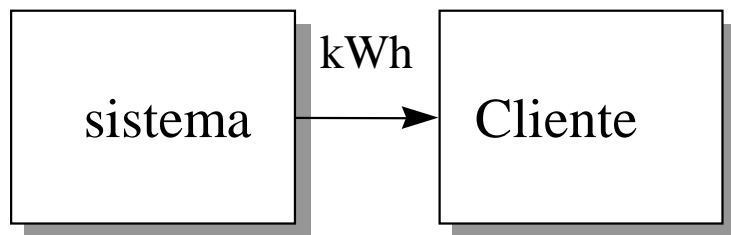


- "Tengo frío. Por favor ,¿podría cerrar la puerta?"
 - La primera parte es informativa: mensaje de intercambio de información.
 - La segunda parte es un requerimiento de una acción para otro agente: mensaje de delegación de tarea.
 - Así, en una única transacción: dos mensajes diferentes en contenido e intención
- Las transacciones no sólo transmiten contenido, sino también una relación pretendida entre dos agentes.
- Los lenguajes de comunicación de agentes (ACL) están inspirados por la "teoría del acto del habla"
- Hace distinciones entre:
 - Contenido ('naturaleza locuacional') de un acto de habla o mensaje--> lo que realmente se está diciendo
 - Efecto pretendido ('fuerza ilocuacional') en el otro agente
 - Efecto real ('fuerza perlocuacional') en el otro agente
- Deben especificarse explícitamente ambos aspectos.

TRANSACCIÓN	Identificador/Nombre Transacción
AGENTES INVOLUCRADOS	Emisor, receptor
ITEMS DE INFORMACIÓN	<p>Lista de todos los items a transmitir . Para cada uno de ellos especificar:</p> <ul style="list-style-type: none"> •Rol: objeto central, objeto de soporte (textos explicativos, fotografías, trazados de razonamiento, etc). •Forma: forma sintáctica en que se transmite (cadenas de datos, diagramas, etc). •Medio: ventana pop-up , interfaz de linea de comandos, menú, intervención humana, etc. <p>(Cuidado: Modelo de diseño)</p>
ESPECIFICACIONES DEL MENSAJE	<p>Descripción de todos los mensajes de la transacción. Para cada uno de ellos:</p> <ul style="list-style-type: none"> •Tipo de comunicación: Tipo de mensaje en cuanto a intención •Contenido: declaración o proposición del mensaje •Referencia: si es necesario, p.ej. indicar qué modelo del dominio o capacidad del agente se requiere para enviar o procesar el mensaje
CONTROL SOBRE LOS MENSAJES	Si necesario, pseudocódigo o diagrama de transición de estados



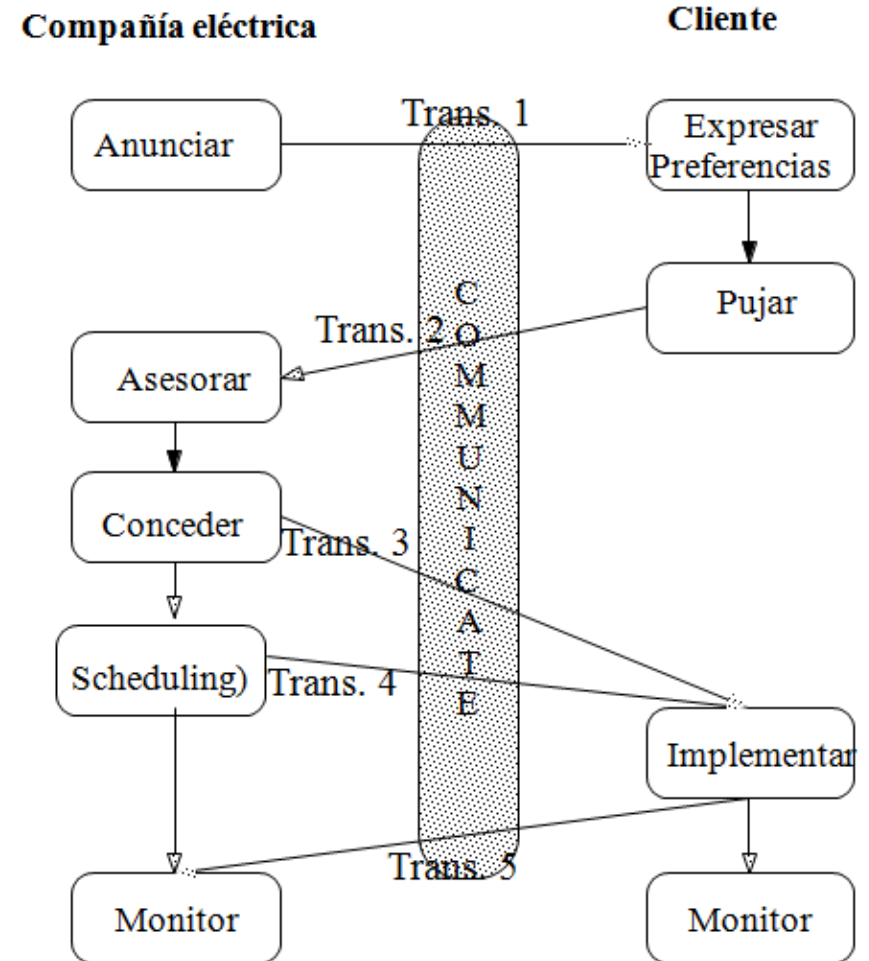
- Mercado energético
 - El precio de la energía varía dinámicamente
 - Permite a las compañías hacer manejo de cargas
 - HOMEBOTS: agentes eléctricos inteligentes
 - Requiere sistema de comunicación de dos direcciones



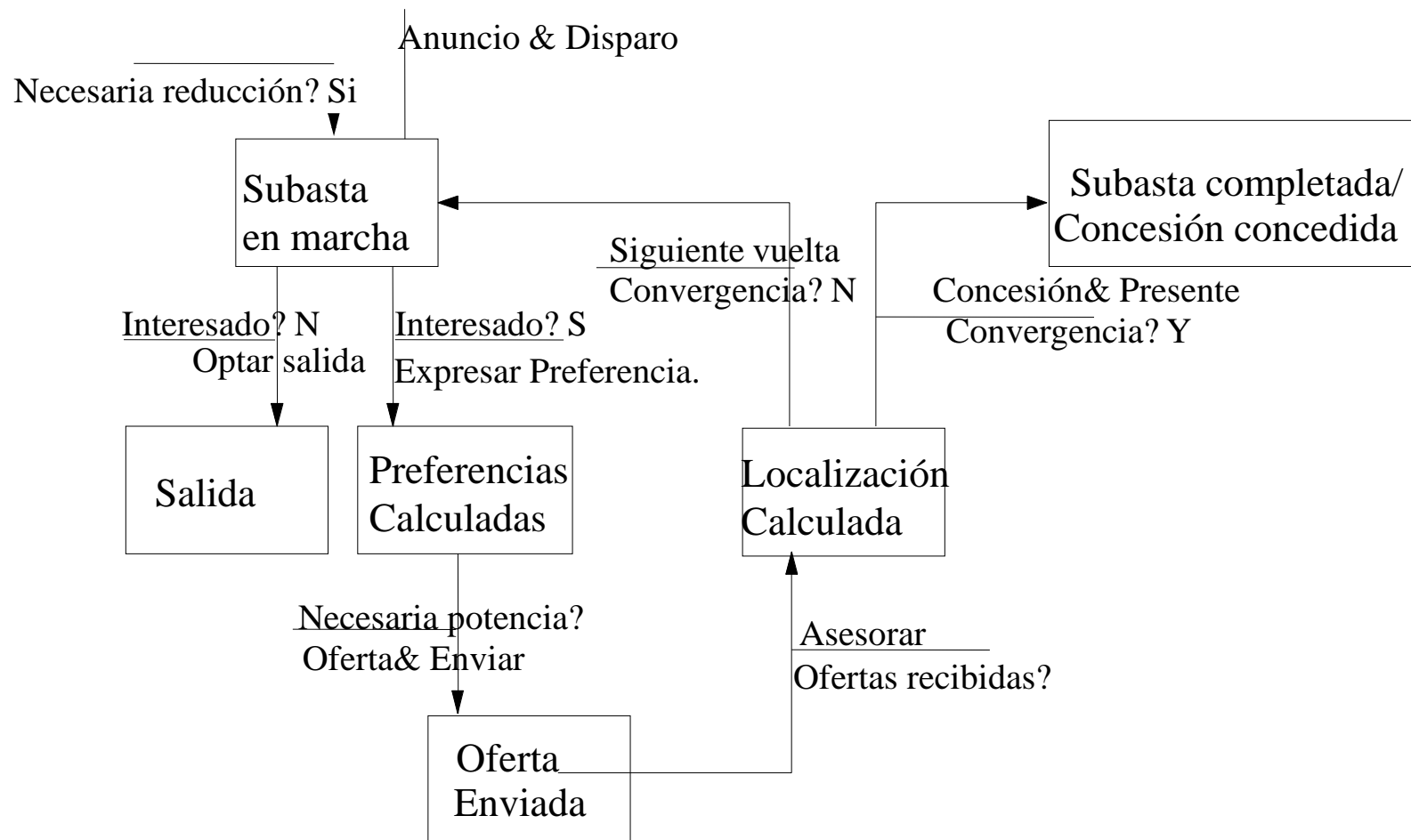


- A través de microprocesadores en red, los aparatos pueden “hablar con”, “negociar”, “tomar decisiones” y “cooperar” unos con otros.
 - Los agentes inteligentes de los equipos se llaman **Homebots** (inspirándonos en Star Trek y los cuentos de Robots de Asimov)
 - Uso por ej. para el manejo distribuido de la carga de potencia
- Beneficios:
 - maneja una escala mucho mayor
 - mayor grado de automatización
 - aproximación descentralizada y flexible
 - proactiva para el cliente

- **Trans. 1. Iniciar la subasta**
- **Trans 2. Enviar la puja**
- **Trans 3. Presentar localización concesión de potencia**
- **Trans 4. Presentar plan asociado en tiempo real**
- **Trans 5. Recibir los datos resultantes de implementación en tiempo real**



Homebots: Control de Diálogo





- Lista de transacciones:
 - **1. Iniciar la subasta:** envía una señal de disparo a los agentes clientes para empezar una acción de manejo de carga
 - **2. Enviar la puja:** transmite las ofertas de los agentes clientes al subastador para su procesamiento posterior.
 - **3. Presentar localización concesión de potencia:** informa a los agentes clientes de los resultados de la subasta
 - **4. Presentar plan asociado en tiempo real:** proporciona a los agentes clientes la lista o plan calculado que implementa la localización ganadora.
 - **5. Recibir datos resultantes de la implementación en tiempo real:** transmite los datos medidos reales (necesarios para facturar y asesorar la necesidad de posteriores acciones de manejo de carga)



Modelo de Comunicación	CM-1: Homebots
TRANSACCIÓN	Trans. 2: Enviar la puja. Transmite las pujas de los agentes clientes al subastador para ser procesadas: El subastador hace los cálculos intermedios (precio y localización) y a su vez las transmite a los agentes clientes para permitir la revisión de la puja.
OBJETOS DE INFORMACIÓN	Unir las tareas <i>Pujar</i> y <i>Asesorar</i> : (1) Pujas de los clientes; (2) cálculos intermedios del subastador (precio o localización)
AGENTES INVOLUCRADOS	(1) agentes clientes, enviando pujas y recibiendo los datos actuales de mercado; (2) agente subastador/sistema, recibiendo pujas y enviando los datos de mercado.
PLAN DE COMUNICACIONES	Homebots (versión base)
RESTRICCIONES	Durante la transacción es necesario un procedimiento de decisión que diga qué hacer cuando los agentes no envían pujas (ej. frecuencia de reintento, tiempo máximo de espera), ya que podría deberse aun fallo de comunicación. Se deriva una post-condición de la condición de convergencia para el equilibrio del mercado (ej. todos los precios a los agentes que pujan deben ser finalmente iguales).
ESPECIFICACIÓN DE INTERCAMBIO DE INFORMACIÓN	Esta transacción contiene más de un mensaje, y por tanto es compuesta. Ver formulario CM-2 correspondiente

Transacción	Transacción 2: Enviar la puja
Agentes involucrados	1.- Emisor (cliente): puja 2.- Receptor (subastador):puja 3.- Emisor (subastador): datos de mercado actuales (precio y localización) 4.- Receptor (cliente): datos mercado actuales
Items de información	Para ambos objetos, pujas y datos mercado: 1.- Rol, ambos son objetos centrales de información. No hay items de soporte en la transacción 2.- Forma, datos numéricos (real). 3.- Medio, no interesa. La interacción entre agentes es automática, con un entorno software estándar y un protocolo de comunicaciones.
Especificaciones de mensaje	<div> 1.- Mensaje puja Tipo: PROPOSE Contenido: Puja Referencia: teoría de mercado De:agentes clientes A: subastador </div> <div> 2.- Mensaje optar-salida Tipo: REJECT-TA Contenido: No más participación en Puja De:agentes clientes A: subastador </div> <div> 3.- Mensaje Datos-subasta Tipo: INFORM Contenido: Datos actuales de mercado Referencia: teoría de mercado De:subastador A: agentes clientes </div> <div> 4.- Mensaje siguiente-vuelta Tipo: REQUEST Contenido: Señal disparo para nueva ronda de subasta De:subastador A: agentes clientes </div>
Control sobre mensajes	Ver pseudocódigo



Control sobre mensajes. Transacción 2. Enviar la puja

REPEAT

WHILE <condición de convergencia de mercado no satisfecha>

IF <interés en manejo de cargas>

THEN PROCESS(tarea-oferta);

SEND(MENSAJE-OFERTA)

ELSE SEND(MENSAJE OPTAR-SALIDA)

END-IF

IF <ofertas recibidas>

THEN PROCESS(tarea-asesorar)

ELSE PROCESS(subproceso decisión [e.g. WAIT...])

END-IF

SEND(MENSAJE DATOS-OFERTAS) &

SEND(MENSAJE SIGUIENTE-VUELTA)

END-REPEAT



- Caminos (walk-throughs) en el plan de comunicaciones. Revisión entre pares (peer to peer).
 - adecuación de la estructura de transacciones
 - completitud de la lista de items de información
 - Necesidad de ayuda o explicación

- Técnica Mago de Oz
 - Técnica experimental para validar la comunicación con un SBC
 - Un experto humano juega el papel del SBC e imita su comportamiento frente al usuario final.
 - Método ideal para comprobar los patrones de pregunta-respuesta

- Usabilidad: Evaluación heurística
 - Una parte significativa del modelo está ligada a temas interfaz de usuario.
 - Mirar temas de usabilidad específicos

- Entradas clave:
 - tareas hoja de TM
 - funciones de transferencia de KM
- Capacidades de los agentes (ver AM) :
comprobar compatibilidad con
restricciones de las transacciones en
el CM.
 - Si CM requiere capacidades
adicionales de agente, añadir en AM.
- Verificar compatibilidad del plan de
comunicaciones con la estructura,
procesos, recursos, etc. del modelo de
organización.
- La regla de conservación de la
estructura aplica en todo el modelado
conceptual.
- Tanto la forma sintáctica como los
medios son un área de ambos el
modelo de comunicación (CM) y el
de diseño (DM)
 - Regla general: Modelar aspectos
de medio y forma en el modelo de
diseño.
 - incluir en CM si hay razones
conceptuales (p.ej firma
electrónica)
- CM (y no DM) debe incluir la
información de soporte (qué items
introducir).



TEMA 2: CommonKADS Modelos Conceptuales