Introducción a la representación del conocimiento

Jordi Duran Cals
Jordi Conesa i Caralt
Robert Clarisó Viladrosa

PID_00199502

UOC
Universitat Oberta de Catalunya

Índice

Introducción

En el material siguiente, se presenta una de las principales áreas de la inteligencia artificial (IA), la representación del conocimiento, que es fundamental para la resolución de gran variedad de problemas. Esta área es muy amplia y alcanza tanto aspectos teóricos como los más puramente epistemológicos, e incluso los problemas más prácticos de tratamiento de datos. Este material se centrará en la vertiente más aplicada: las estrategias que permiten codificar el conocimiento humano para ser utilizado computacionalmente. De hecho, pese a enfocarse en el área de la IA, los principios de la representación del conocimiento se utilizan en otras subáreas de la informática como, por ejemplo, la modelización conceptual o las bases de datos.

Epistemología

La epistemología es la ciencia que reflexiona filosóficamente sobre el proceso del conocimiento humano y los problemas que se presentan en el mismo.

Tal y como indica su propio nombre, la representación del conocimiento tiene la intención de definir mecanismos para representar el conocimiento por medio de símbolos y facilitar de este modo su tratamiento. Estos mecanismos son los encargados de nutrir sistemas inteligentes dirigidos a resolver tareas complejas que requieren razonar sobre un determinado dominio de aplicación. Muchas veces, la eficacia de los sistemas inteligentes depende de una buena representación de los datos, de la información o del conocimiento del dominio en el cual actúen. Por lo tanto, estamos ante un área clave en el campo de la ingeniería informática.

Antes de empezar, hay que tener presente que la tarea de modelizar la realidad es una tarea imposible con las técnicas y herramientas actuales. La realidad tiene demasiadas variables, casos particulares y relaciones causales como para describirla totalmente. De hecho, representar de manera eficiente tan solo una parte de la realidad, la relevante para un problema concreto, ya es un trabajo de unas dimensiones astronómicas. En este material mostraremos algunos proyectos de investigación que llevan décadas persiguiendo este objetivo tan ambicioso.

Objetivos

Con el estudio de este módulo didáctico, alcanzaréis los objetivos siguientes:

1. Saber analizar el conocimiento por medio de la clasificación de su procedencia y forma.
2. Conocer la complejidad y los límites de representar el conocimiento.
3. Conocer con detalle diferentes representaciones del conocimiento.
4. Saber exponer los conjuntos de esquemas de representación y sus principales aplicaciones.
5. Saber qué formalismo específico es más apropiado para representar cada tipo de conocimiento.

1. El conocimiento

El término *conocimiento* se utiliza de manera cotidiana con un significado muy genérico.

Conocimiento

"Hechos, información y habilidades adquiridas mediante la experiencia o el aprendizaje para la comprensión teórica o práctica de un tema. También, conciencia o familiaridad adquirida por la experiencia de un hecho o una situación." (*Oxford Dictionary*.)

En este apartado, estudiaremos el papel del conocimiento en un ámbito concreto de la informática, la inteligencia artificial, y consideraremos los diferentes tipos de conocimiento que intervienen en el mismo.

1.1. El conocimiento en la inteligencia artificial

Uno de los objetivos de la inteligencia artificial (IA[1]) es el desarrollo de técnicas y métodos que permitan a un sistema informático resolver problemas de manera inteligente, es decir, teniendo en cuenta el contexto y la información disponible para alcanzar el objetivo deseado. Resolver este tipo de problemas puede requerir determinadas habilidades como la capacidad de aprender, de razonar o de planificar, entre otras.

(1)IA es el acrónimo de inteligencia artificial.

Un requerimiento implícito en el desarrollo de un sistema inteligente es la capacidad de representar y utilizar el conocimiento necesario para resolver el problema tal y como lo haría un experto humano.

Ejemplo 1. Conducción de vehículos terrestres: tipo de conocimiento requerido

Una aplicación práctica de la inteligencia artificial es el desarrollo de sistemas capaces de conducir vehículos terrestres sin la supervisión de los humanos. Un sistema como este tiene que ser capaz de tratar e interrelacionar conocimiento de diferentes ámbitos.

a) Conocimiento sobre el propio vehículo:

- Sus características (altura, anchura, longitud, peso, etc.).
- Su estado (velocidad, combustible disponible, número de ocupantes, carga total, posibles averías, etc.).

b) Conocimiento sobre el problema de la conducción:

- Las normas del código de circulación.
- Las leyes físicas relacionadas con el movimiento, la aceleración, etc.

Observación

En este punto, y hasta que presentemos una definición más precisa del término *conocimiento*, abusaremos del lenguaje para simplificar la lectura y utilizaremos la palabra *conocimiento* cuando en realidad nos referimos a datos o a información.

c) Conocimiento sobre el entorno:

- Los otros vehículos de la vía (posición, dirección y velocidad).
- Otros objetos relevantes para la conducción (carriles, señales de tráfico, obstáculos en la ruta, etc.).
- Las condiciones climatológicas (lluvia, nieve, niebla, etc.).
- Las condiciones de la carretera (iluminación, tipo de vía, incidencias, etc.).

Desde que en 1956 aparece el término IA en la *Darmouth Conference*, hasta hoy día, los profesionales de esta área han construido sistemas que incorporan conocimiento de manera más o menos explícita. Muchos sistemas inteligentes utilizan de modo intensivo el conocimiento de algún **dominio**, el campo de aplicación en el que definimos nuestro problema. El abanico de ejemplos es muy grande y variado, y tenemos como muestra sistemas de diagnosis médica y de análisis lingüístico. Sin embargo, en muchos casos el conocimiento que contienen no está representado explícitamente, algo que no lo hace ni extensible ni manipulable.

Los dominios de discurso

Un dominio de discurso (o simplemente, dominio) es un área de conocimiento que engloba el conocimiento relevante que queremos representar. Ejemplos de conocimiento pueden ser el subconjunto de la física dinámica necesaria para entender la conducción de los vehículos, información sobre las funciones de un vehículo, sobre su interfaz, etc.

Ejemplo 2. Jugar al ajedrez: el uso de la fuerza bruta frente al uso de conocimiento

Un programa informático que juega al ajedrez puede intentar encontrar la mejor jugada en cada posición utilizando únicamente "la fuerza bruta": probando un gran número de combinaciones de jugadas. No obstante, el programa resultará mucho más efectivo si es capaz de reconocer y aprovechar conceptos del ajedrez como por ejemplo el de *apertura*, *final de rey y peones* o *jaque mate con torre y rey*. En estos casos, y aprovechando este conocimiento, el programa podría ajustar su proceso de búsqueda o bien utilizar una biblioteca de jugadas predefinidas.

En consecuencia, la comunidad científica ha reconocido la necesidad de **modelizar el conocimiento**.

Modelizar el conocimiento

"Si el propósito de trabajar con la IA es modelizar la mente humana o diseñar sistemas inteligentes, necesariamente incluye un estudio del conocimiento." (Aaron Sloman, 1979.)

Como veremos a continuación, dentro del campo de la IA hay diferentes concepciones sobre cómo utilizar el conocimiento para la resolución de problemas.

1.2. IA simbólica frente a IA subsimbólica

La disciplina de la IA se clasifica en un conjunto de áreas, como por ejemplo la resolución de problemas y búsqueda, la inteligencia artificial distribuida, la representación del conocimiento y el aprendizaje computacional[2]. Las dos

[(2)]En inglés, *machine learning*.

últimas áreas están estrechamente relacionadas entre sí y representan dos paradigmas diferenciados en el campo de la IA: la **IA simbólica** y la **IA subsimbólica**.

a) IA simbólica

En la IA simbólica, el conocimiento se representa por medio de unidades discretas (símbolos) que se pueden combinar siguiendo ciertas normas en un formalismo o lenguaje de estructuras más complejas: propiedades, fórmulas, reglas, relaciones, sentencias, etc.

Ejemplo 3. Norma del dominio de circulación viaria

En un sistema de conducción automática, el código de circulación viaria es crucial para circular de manera segura y legal. Este conocimiento está descrito como un listado de normas, cada una de las cuales se puede representar de manera simbólica. Un ejemplo de esto podría ser la regla siguiente.

```
Si circulamos por una vía en EE. UU. entonces hay que circular por
la derecha
```

Históricamente, la IA simbólica es la aproximación original hacia la IA que se conoce como IA clásica. El campo de la representación del conocimiento se engloba dentro de este paradigma, y será el objeto de estudio de este módulo.

Reflexión

En el apartado 2 de este módulo, estudiaremos diferentes formalismos de representación, mediante los cuales explicaremos, entre otros aspectos, las representaciones lógicas, las redes semánticas o los marcos (*frames*).

b) IA subsimbólica

En la IA subsimbólica, el conocimiento se transmite por medio de propiedades implícitas de los objetos.

Este tipo de representación, utilizada en el área de aprendizaje computacional, aparece como consecuencia de las limitaciones de la IA clásica: en algunos problemas, los formalismos existentes son inapropiados para capturar toda la información requerida con el nivel de detalle necesario. Esto sucede, por ejemplo, en problemas en los que se trabaja con conocimiento incierto, relaciones espaciales, etc. En otros casos, el problema es la falta de un experto del dominio o estar ante un dominio de dimensiones considerables y que hace inviable representarlo de manera simbólica.

Ejemplo 4. Reconocimiento facial: ejemplo de IA subsimbólica

Un problema típico en visión por computador es identificar un objeto, una persona o una ubicación a partir de una imagen. Aunque un objeto tenga una forma muy característica, reconocerlo puede llegar a ser un problema muy complicado. Puede variar el ángulo de visión, la distancia a la que nos encontramos o el nivel de iluminación, o bien quizá haya obstáculos que lo oculten de manera parcial. Así pues, no es posible describir simbólicamente todas las distintas formas en las que podemos percibir un objeto en una imagen.

En estas situaciones, la solución pasa por utilizar técnicas de aprendizaje basadas en ejemplos: máquinas de soporte vectorial[3], redes neuronales, etc. Estas técnicas analizan un conjunto de ejemplos para calcular pesos, probabilidades o heurísticas que se utilizarán en el proceso de decisión. Es decir, intentan extrapolar el conocimiento a partir de los ejemplos, aunque este conocimiento no se acaba explicitando sino que queda implícito en los valores calculados.

(3)En inglés, *support vector machines*.

Ejemplo 5. Redes neuronales: ejemplo de IA subsimbólica

Una red neuronal es un modelo matemático de inspiración biológica que imita el funcionamiento de las neuronas del sistema nervioso. En lugar de asignar una tarea diferenciada a cada neurona, el comportamiento de la red está determinado por las conexiones entre las neuronas.

En la fase de entrenamiento, el peso o la relevancia de cada conexión se configura a partir de los ejemplos disponibles. Una vez acabado el entrenamiento, estos pesos se utilizarán para dar respuesta ante nuevos escenarios.

De este modo, una red neuronal puede "aprender" a reconocer formas (caras, señales de tráfico, etc.) a partir de ejemplos. Sin embargo, no podrá "explicar" de manera simbólica el porqué de sus decisiones, puesto que esta información queda implícita en los valores de los pesos de las conexiones. Esto hace muy difícil manipular o ampliar este conocimiento.

Campos de aplicación de la IA

Los campos más comunes en los que se aplican métodos de IA subsimbólica son la visión por computador, la robótica y el procesamiento del lenguaje natural, entre otros.

Ejemplo 6. Algoritmos genéticos: ejemplo de IA subsimbólica

Los algoritmos genéticos son un esquema de búsqueda también de inspiración biológica. Ante el reto de encontrar la mejor solución para un problema según un cierto criterio de optimización, este método intenta imitar el proceso de evolución y selección natural.

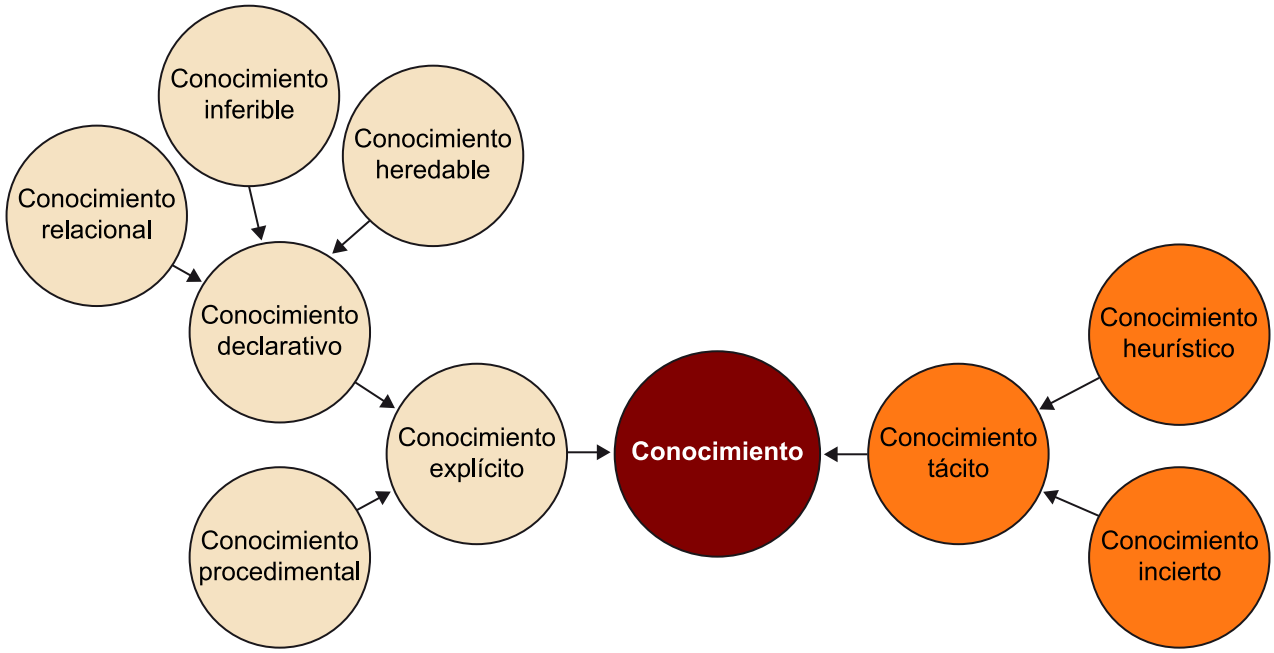
El esquema parte de una población en la que cada individuo es una solución potencial. En cada etapa de la búsqueda (generación), se evalúa la calidad de las soluciones y se elimina una parte de las soluciones menos prometedoras, manteniendo las mejores. Además, en la nueva generación se introducen **mutaciones** en forma de cambios en soluciones existentes y **cruces** entre soluciones distintas para generar nuevos individuos. El proceso continúa durante un cierto número de iteraciones y, finalmente, se selecciona la mejor solución encontrada.

Aunque este proceso puede conseguir soluciones muy buenas a problemas de búsqueda complejos, no permite extraer conocimiento del proceso. Por ejemplo, no permite explicar qué factores hay que tener en cuenta para construir buenas soluciones, puesto que la búsqueda tiene un componente muy elevado de aleatoriedad.

1.3. Clasificación del conocimiento

En el transcurso de centenares de años de estudio del conocimiento en campos como la filosofía o la religión, se han considerado numerosas clasificaciones posibles. A continuación, presentamos de manera gráfica una clasificación de los tipos de conocimiento que pueden aparecer en la modelización de un dominio. A lo largo de los subapartados siguientes, expondremos sus características.

Figura 1. Tipos de conocimiento y sus relaciones



1.3.1. La jerarquía del conocimiento

Para estudiar de dónde procede y cómo se obtiene el conocimiento, algunos teóricos como J. Rowley (2007) utilizan la relación jerárquica entre los conceptos de datos, información, conocimiento y sabiduría (*DIKW*[4]).

(4)Del inglés *data, information, knowledge and wisdom.*

La jerarquía del conocimiento empieza en el nivel más bajo a partir de unos **datos**, de los que obtenemos **información** al organizarlos y analizarlos. A continuación, la interpretación o evaluación de la información nos da el **conocimiento**. Finalmente, la **sabiduría** es la comprensión de los principios que el conocimiento alcanza interiormente.

Ahora podremos ver con más detalle cada uno de estos niveles.

a) Datos

Los datos son valores, hechos y evidencias sobre un aspecto concreto de un objeto o concepto.

Ejemplo 7. Significado de datos

Una señal de tráfico puede tener forma geométrica de octágono regular, triangular, redonda o cuadrada. Por ejemplo, un conjunto de datos sobre una señal podrían ser una señal de color rojo con la forma geométrica de un octágono regular y con el texto "STOP" inscrito en su interior.

b) Información

La información es una idea elaborada sobre un objeto o concepto a partir de la combinación de sus datos.

Ejemplo 8. Significado de información

Una señal de stop tiene la forma geométrica de un octágono regular con fondo de color rojo y la palabra "STOP" inscrita en mayúsculas en color blanco. Por lo tanto, la información sería discernir que la señal descrita en el ejemplo anterior es una señal de stop.

c) Conocimiento

El conocimiento es información contextual que ayuda a entender una situación real y define la experiencia.

Ejemplo 9. Significado de conocimiento

Si encontramos una señal de stop en el sentido de la circulación, entonces tenemos que detener el vehículo, buscar una buena visibilidad y ceder el paso, si procede, antes de continuar. Este conocimiento se crea a partir de la información que tenemos: hay una señal de stop, está en la dirección de la marcha y la información de que la señal de stop controla la preferencia de paso de diferentes vehículos en un cruce.

d) Sabiduría

La sabiduría es el peldaño más elevado de la comprensión, es decir, la comprensión completa de los efectos y los resultados del conocimiento.

Ejemplo 10. Significado de sabiduría

No detenerse en un stop supone un riesgo muy elevado de sufrir un accidente. Por precaución, aunque el stop afecte a otros vehículos, es importante estar alerta y comprobar que han visto el stop y se han parado. Este razonamiento parte del hecho de que las señales de stop suelen situarse en cruces donde el conductor tiene poca visibilidad o poco margen de reacción.

1.3.2. Conocimiento tácito frente a conocimiento explícito

Un primer criterio de clasificación del conocimiento es la posibilidad de comunicarlo. Según este criterio, podemos distinguir entre el conocimiento tácito (o implícito) y el conocimiento explícito:

Observación

Observad que hay un paralelismo claro entre el conocimiento explícito y tácito y la IA simbólica y subsimbólica.

Tabla 1

Conocimiento tácito	Conocimiento explícito
• Existe encarnado dentro del ser humano.	• Puede existir fuera de los seres humanos.
• Resulta difícil de expresar en una estructura.	• Es fácil de estructurar y expresar.
• Es complicado de comunicar o compartir.	• Puede ser compartido, procesado y guardado.
• Se elabora a partir de la experiencia, los actos y también una visión subjetiva.	• Se elabora a partir de conceptos, procesos, procedimientos y principios.

Ejemplo 11. Conocimiento tácito y conocimiento explícito

Un ejemplo de conocimiento tácito es el reconocimiento facial. Las personas tenemos la capacidad de reconocer la cara de una persona conocida entre un millón, aunque no sepamos describir con palabras cómo lo sabemos.

Por el contrario, un ejemplo de conocimiento explícito es una ruta entre dos puntos: es factible dar un mapa y un listado de indicaciones para ir desde el origen hasta el lugar de destino.

Obviamente, en el estudio de la representación del conocimiento nos centraremos en el conocimiento explícito, puesto que es el que podemos codificar y comunicar.

En el proceso de creación del conocimiento explícito, están involucrados los objetos siguientes:

a) Hechos: instancias o datos específicos y únicos.

Ejemplo 12. Hechos

Ejemplos de hechos podrían ser el número de accidentes que ha habido en el 2012 o la constatación de que uno de los precursores de la informática fue Alan Turing o que el color de fondo de una señal de stop es rojo.

b) Objetos o individuos: elementos concretos del mundo real, entendiendo como elementos cualquier objeto concreto (personas, animales, edificios, documentos, etc.) o abstractos (tiempo, hora, números, palabras, etc.). Los individuos pueden ser tangibles o no. En algunos ámbitos, también se utiliza el término *instancia* para referirse a individuos/objetos.

Observación

A partir de aquí, utilizaremos los términos *objeto*, *instancia* e *individuo* de manera indistinta para referirnos al mismo concepto.

Ejemplo 13. Instancias

Ejemplos de individuos serían el número 3, el comandante Spock o Leonard Nimoy (que es el actor que lo representa).

c) Conceptos o clases: clases de elementos, palabras o ideas que son conocidas por un nombre común y comparten características también comunes. Los conceptos permiten clasificar objetos en función de sus características comunes, como por ejemplo la clase *Estudiantes de la asignatura de "Representación del conocimiento"*, que incluiría a todos los estudiantes de esta asignatura. Otros ejemplos de clase serían la clase *Actores*, la clase *Estudiantes*, la clase *Facturas*, etc.

d) Relaciones entre clases: indican posibles relaciones entre las clases o entre sus instancias. El número de elementos que permite relacionar una relación suele ser fijo y viene determinado por su aridad. Aunque es posible tener relaciones de aridad superiores a 2, es difícil encontrar representaciones que utilicen relaciones con una aridad de 4 o superiores. Las relaciones acostumbran a tener una semántica que indica cuál es la relación entre los objetos relacionados y un conjunto de restricciones de integridad. Las restricciones de

integridad pueden ser muy variadas, pero por norma general permiten indicar cuántos objetos pueden participar en la relación, de qué tipo tienen que ser estos objetos y su obligatoriedad de participar en la relación.

Ejemplo 14. Relaciones

Algunos ejemplos de relaciones serían los siguientes:

- La relación denominada *estudia en*, que relaciona la clase *Estudiantes* con la clase *Asignaturas* y permite representar las asignaturas que cursa cada estudiante.

- La relación denominada *madre biológica*, que relaciona la clase *Mujer* con la clase *Persona*. En este caso, se podría añadir una restricción de integridad para indicar que todo humano debe tener por fuerza una madre y solo una, para lo que sería necesario restringir el número de madres que puede tener una persona (a una) y definir la obligatoriedad de que toda persona participe en la relación (toda persona tiene madre).

- La relación *ha obtenido medalla* entre las clases *Atleta*, *Juegos Olímpicos*, *Tipos de medalla* y *Prueba*, que permite representar las medallas que obtienen los atletas en unas pruebas de unos Juegos Olímpicos, así como de qué tipo son las medallas obtenidas. Como se puede comprobar, esta relación sería cuaternaria, es decir, tendría una aridad de 4.

e) Principios: directrices, reglas que se tienen que satisfacer, normas y parámetros que rigen y permiten tomar decisiones y extraer consecuencias. Los tipos de principios más conocidos son las restricciones de integridad, las reglas de derivación y las reglas heurísticas:

- Las restricciones de integridad definen condiciones que todas las instancias deben satisfacer para ser válidas. Ejemplos claros de restricción de integridad son "la edad de una persona no puede ser negativa" o "un hombre no puede ser la madre biológica de una persona".

- Las reglas de derivación permiten calcular de manera automática instancias (o valores de estas) a partir de otra información. La regla que permite calcular la edad de una persona a partir de su fecha de nacimiento sería un ejemplo de regla de derivación.

- Las reglas heurísticas permiten indicar condiciones que se cumplen en muchos casos, pero que no se satisfacen siempre. Ejemplos de reglas heurísticas serían "los pianos son grandes" o "los animales de compañía son apacibles".

f) Eventos (**procesos**/**procedimientos**): indican acciones que se pueden llevar a cabo sobre el conocimiento descrito o eventos que pueden suceder en el dominio de discurso. Pueden definirse de manera declarativa o procedimental. En el primer caso se definirá cuál es el resultado esperado, mientras que en el segundo caso se definirán las acciones y decisiones necesarias para resolver una tarea y en qué orden se tienen que ejecutar (es decir, cómo llegar al resultado esperado).

1.3.3. Conocimiento declarativo frente a conocimiento procedimental

Un segundo criterio para clasificar el conocimiento es el tipo de pregunta al que da respuesta: "¿qué?" o "¿cómo?". Según este criterio, podemos distinguir entre conocimiento declarativo (o descriptivo) y conocimiento procedimental (u operacional).

El **conocimiento declarativo** se centra en la representación de los hechos o las afirmaciones, es decir, nos indica qué es cierto o falso refiriéndose a los objetos y los eventos.

Símil

Haciendo un símil, podemos ver el conocimiento declarativo como una base de datos de alto nivel.

Desde el punto de vista de la descripción del mundo, el conocimiento declarativo es un **conocimiento factual**. Es decir, estructura los hechos, los significados, los conceptos y el conocimiento sobre el mundo externo que las personas podemos adquirir. Además, se puede compartir porque es independiente de la experiencia personal y del contexto espacial y temporal.

Así pues, el conocimiento declarativo se limita a enunciar información pero sin tener en cuenta los procesos ni algoritmos que se aplicarán. Esto es muy útil para facilitar la agregación de nuevo conocimiento en la estructura que lo sustenta, pero también nos limitará en el proceso de inferencia. Hay que decir, sin embargo, que se trata posiblemente del conocimiento más utilizado y, al mismo tiempo, el más conocido.

Ejemplo 15. Conocimiento declarativo en el contexto de conducción de vehículos

Un coche está bien aparcado cuando se encuentra a una distancia prudencial de los vehículos circundantes y de las aceras, sin tocar ninguno de estos elementos, y no infringe ninguna norma viaria.

Observad que el conocimiento descrito es declarativo porque describe qué significa aparcar bien, pero no cómo hacerlo.

Los tipos de conocimiento declarativo son el relacional, el inferible y el heredable. Antes de analizarlos en detalle, estudiemos primero cómo es el conocimiento procedimental.

El **conocimiento procedimental**[5] nos indica cómo se hace algo, es decir, se centra en las acciones que hay que llevar a cabo para lograr un objetivo concreto.

(5)Enfocado a la tarea para llevar a cabo un objetivo.

Podemos ver el conocimiento procedimental como un conjunto de pasos e instrucciones que especifican el conjunto de acciones para llevar a cabo una tarea. Como ventaja con respecto al conocimiento declarativo, es un tipo de conocimiento muy fácil de aplicar, pero por otro lado, se hace más difícil la inferencia, la modificación y la verificación.

Ejemplo 16. Conocimiento procedimental

Es posible conocer toda la teoría para la conducción de un coche; por ejemplo, qué pedal es el acelerador, qué direcciones toma el cambio de marcha o qué significan las señales de tráfico, pero estos conocimientos declarativos no nos sirven para saber conducir un coche. El conocimiento de la conducción es procedimental, muy distinto a conocer una colección de hechos.

También relacionado con el ejemplo anterior, el conocimiento que explica cómo estacionar un coche es procedimental: comprobar que está permitido estacionar, situarse en paralelo al coche anterior adyacente al lugar de estacionamiento, introducir el vehículo en el hueco dando marcha atrás, etc.

A diferencia del conocimiento declarativo, el conocimiento procedimental incluye también los procesos para utilizar y manipular el conocimiento declarativo. Los mecanismos para representar el conocimiento procedimental los tenemos en los programas que todos conocemos y que podemos ver como algoritmos, reglas, estrategias y modelos.

En los apartados siguientes veremos con más detalle los tres tipos de conocimiento declarativo (relacional, inferencial y heredable), así como el conocimiento lingüístico.

1.3.4. Conocimiento relacional

La manera más simple y habitual de representar hechos declarativos consiste en un conjunto de relaciones expresadas en forma de tablas, al igual que se haría en una base de datos.

Ejemplo 17. Conocimiento relacional

Como ejemplo de conocimiento relacional mostramos la tabla 2, que indica que los conductores tienen los atributos *Nombre*, *Edad* y *Tipo de licencia* y permite representar instancias de estos:

Tabla 2. Características del conocimiento relacional

Driver Name	Age	Licence Type
John Smith	25	B
Peter Clinton	32	A

Por sí misma, una tabla solo aporta aquella información (conocimiento) que se pueda extraer directamente, como por ejemplo, saber qué licencia de conducción tiene una persona en concreto.

Por otro lado, podemos añadir procedimientos que enriquezcan el conocimiento. Por ejemplo, podemos añadir un motor de inferencia y así generar nuevo conocimiento a partir del existente y validar el que ya hay. Así pues, nos puede interesar saber cuántas licencias de conducción hay de un tipo determinado, cuál es la media de edad de los conductores de una licencia, etc. También podremos llevar a cabo validaciones de los datos existentes; por ejemplo: "ningún conductor mayor de 75 años puede tener una licencia de tipo A".

Base de datos tradicionales como método de representación

Los sistemas de bases de datos son por regla general muy útiles para la computación, pero en lo que respecta a la IA resultan limitados. No hay duda de que se trata de una manera eficiente de representar y procesar grandes cantidades de datos. Pero entonces estamos limitados a describir solo hechos simples de dominio y a efectuar razonamientos sencillos basados en la búsqueda.

1.3.5. Conocimiento inferencial

El conocimiento inferible o deductivo describe el conocimiento mediante la lógica tradicional. Por lo tanto, utiliza la semántica de los operadores y el *modus ponens* para inferir de nuevo.

Ejemplo 18. Conocimiento inferencial en el caso de la conducción de vehículos

Definiremos el coche de John utilizando la lógica de primer orden:

1. El *Porsche de John* es una instancia de la clase *Coche*:

$$car(\textit{Porsche de John})$$

2. Todos los coches son vehículos:

$$\forall\, x{:}car(x) \rightarrow vehicle(x)$$

3. Todos los vehículos pueden ir por tierra, mar o aire:

$$\forall\, x{:}vehicle(x) \rightarrow travel(x, \text{land}) \vee travel(x, \text{water}) \vee travel(x, \text{air})$$

De estos enunciados, por ejemplo, podemos inferir que el Porsche de John puede ir por tierra, mar o aire. Podemos ver que cuanta más información tengamos de estos objetos, más conocimiento podremos inferir.

1.3.6. Conocimiento heredable

La representación heredable es una estructura jerárquica que permite representar el conocimiento de manera incremental. Se obtiene relacionando clases con otras clases de carácter más general, lo que permite reutilizar el conocimiento general en las nuevas clases y heredar de manera total o parcial el conjunto de atributos que tienen definidos. Con este objetivo, se utiliza una estructura de árbol (taxonomía) o grafo (red) en la que los vértices serán clases –también denominadas **categorías**– y las aristas representarán las **relaciones** entre clases.

Una **categoría** es una abstracción que permite representar las características que tienen en común un conjunto de objetos. Un ejemplo de categoría sería la clase *Coche*, que permite agrupar vehículos terrestres a motor de cuatro ruedas. El concepto de categoría es vital para la representación del conocimiento.

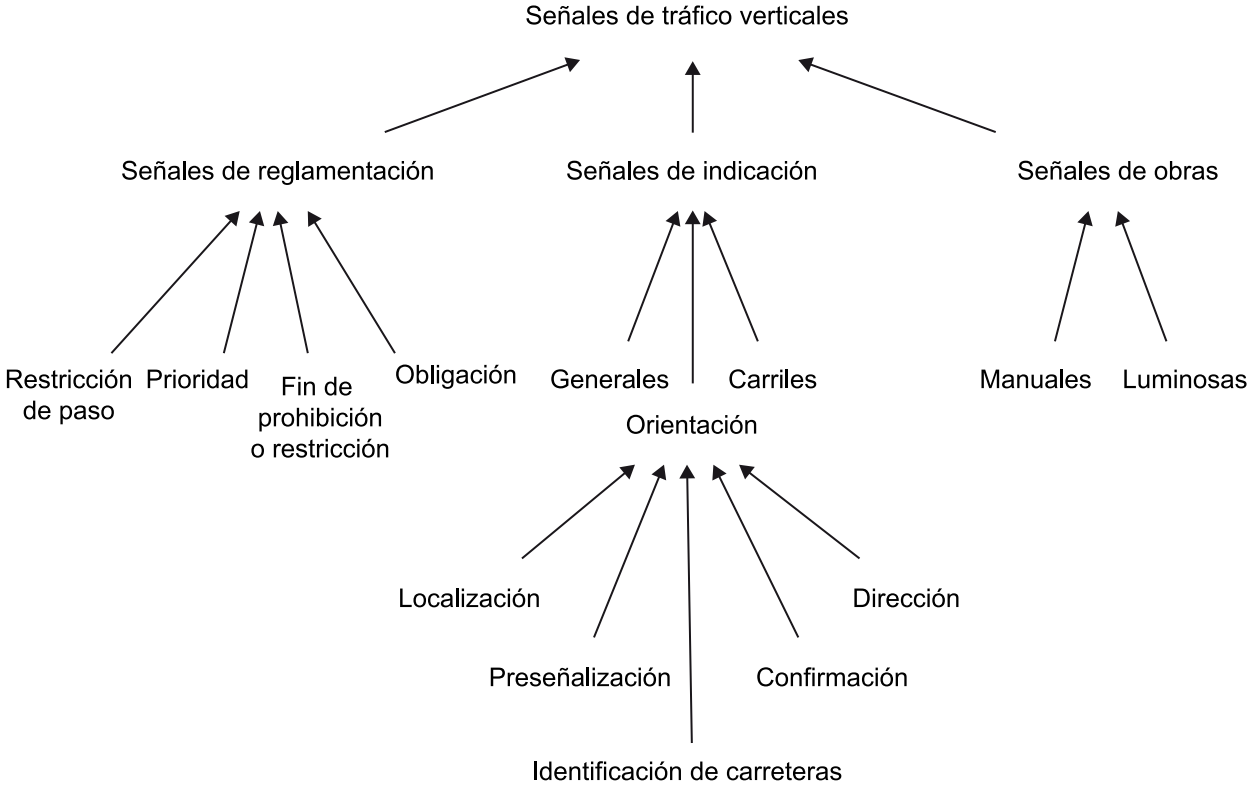
> Aunque la interacción con el mundo se efectúa en un ámbito de objetos individuales, la mayor parte del proceso de razonamiento tiene lugar en el ámbito de las categorías (S. Russell, P. Norvig, 2004).

Si organizamos las categorías en subcategorías por medio de un árbol que las relacione, obtendremos lo que se denomina una **taxonomía**. Conviene destacar que la taxonomía es una estructura utilizada durante siglos en muchos campos técnicos para la clasificación. Uno de los campos es la biología, en la que durante mucho tiempo se ha trabajado para crear una taxonomía para la clasificación de todas las especies vivas y también extinguidas.

Ejemplo 19. Taxonomía de señales de tráfico

En la siguiente imagen, podemos ver una clasificación taxonómica de los diferentes tipos de señales de tráfico verticales. Por ejemplo, el tipo de señal *Señales de dirección* hereda las propiedades de su nodo padre y este al mismo tiempo del suyo, de modo que obtiene la propiedad de ser una señal de orientación e indicación.

Ilustración 2. Ejemplo de taxonomía



Una manera de representar fácilmente las categorías, su relación con otras categorías y los objetos que clasifican es utilizar lógica de primer orden y la teoría de conjuntos, como podemos ver a continuación:

- El operador de pertenencia de elementos a un conjunto se puede utilizar para indicar que un objeto es un miembro de una categoría.

Ejemplo 20. Definición de pertenencia de un objeto a una categoría

La señal de indicación de salida 15 de autopista AP7 es un objeto que pertenece a la categoría *Señales de dirección*:

$$\textit{Señal de salida 15 de la autopista AP7} \in \textit{Señales de dirección}$$

- El operador de subconjunto se puede utilizar para indicar que una categoría es subcategoría de otra. Matemáticamente, *is a* corresponde a la relación de pertenecer a un subconjunto ($\subseteq$), e *instance of*, a la relación de miembro de un conjunto ($\in$).

Ejemplo 21. Definición de subconjunto entre categorías

El conjunto de *Señales de dirección* es un subconjunto de las *Señales de orientación*:

$$\textit{Señales de dirección} \subseteq \textit{Señales de orientación}$$

- La lógica de primer orden se puede utilizar para indicar las propiedades de los objetos de una categoría.

Ejemplo 22. Definición de las propiedades de la categoría *Señal*

Todas las señales están geográficamente ubicadas en algún lugar y, por lo tanto, tienen un atributo para indicar su longitud y otro para indicar su latitud:

$$x \in \textit{Señal} \Rightarrow \textit{latitud}(x, \textit{lat})$$
$$x \in \textit{Señal} \Rightarrow \textit{longitud}(x, \textit{lon})$$

Propiedades o atributos

Una propiedad (o atributo) es un caso particular de relación binaria en el que uno de los participantes es un tipo de datos. Podéis ver el subapartado 2.10.2 de este módulo para más información.

La herencia es la manera de definir las relaciones de pertenencia entre categorías y, por lo tanto, permite definir que una categoría es subtipo de otra categoría. Conceptualmente, la herencia se utiliza para especificar categorías más específicas y que, de este modo, clasifican menos objetos. Desde un punto de vista formal, la definición de la herencia entre dos categorías (que denominaremos categoría y subcategoría) define una restricción de integridad de pertenencia entre las dos categorías, que indica que todo objeto clasificado por subcategoría también debe ser clasificado por categoría. Así pues, si definimos que la categoría *Hombre* es una subcategoría de *Persona*, estaremos creando un esquema que garantizará que todo hombre (por ejemplo, Jorge García) es también una persona (Jorge García está clasificado también como *Persona*). Puesto que una subcategoría define de manera más concreta los elementos de una categoría, todas las relaciones de la categoría son también válidas en la subcategoría (y por lo tanto, aplicables).

Aunque se permite que una subcategoría herede de más de una categoría (lo que se denomina herencia múltiple), a la hora de la verdad no es algo muy común en las representaciones que podemos encontrar. No porque no sea útil ni porque no represente la realidad de manera fiel, sino porque muchos lenguajes de representación no permiten la herencia múltiple. Si no hay herencia múltiple, la jerarquía de categorías y subcategorías se puede representar en

forma de árbol. En este árbol, las instancias de las subcategorías se propagan hacia su categoría (hacia arriba en el árbol) y las relaciones de las categorías se propagan hacia sus subcategorías (hacia abajo en el árbol). Obviamente, si no hay herencia múltiple, la estructura tiene que ser la de un árbol, puesto que por la definición de la herencia no puede haber ciclos.

La generalización y especialización del conocimiento

La manera más común de representar la herencia entre conceptos es mediante relaciones de generalización y/o especialización. Una relación de generalización/especialización permite definir que hay una clase general –que denominamos entidad superclase (o supertipo o padre)– que se especializa en una (o más) clases, denominadas subclases, subtipos o hijas:

1) La superclase define conocimiento más general y acostumbra a definir las características comunes de todas sus subclases.

2) Las subclases definen información más concreta y suelen definir características que no son comunes para todas las subclases de superclase.

Formalmente, podríamos decir que necesitamos dos relaciones para definir la herencia: una para decir que un concepto es más general que otro (relación de generalización), y la otra para decir que un concepto es más específico que otro (relación de especialización). La relación de generalización/especialización es bidireccional y engloba estas dos semánticas. De hecho, se usarán los términos especialización o generalización en función de si se lee la relación de los supertipos a los subtipos o en sentido contrario.

También es importante notar que la herencia, y por lo tanto la generalización y especialización, son aplicables a las relaciones.

Ejemplo 23. Definición de herencia entre relaciones

La relación *tiene una propiedad* que relaciona las clases *Persona* y *Vivienda* e indica que una persona es propietaria de un inmueble puede especializarse en dos relaciones más específicas: *tiene primera residencia* y *tiene segunda residencia*.

Los términos *es un* (*is a* en inglés) y *tipo de* (*kind of* o *type of* en inglés) también se suelen utilizar en algunos lenguajes de representación de conocimiento para representar relaciones de generalización/especialización. Por ejemplo: "Canario es un Animal" o "Silla es un tipo de Mueble".

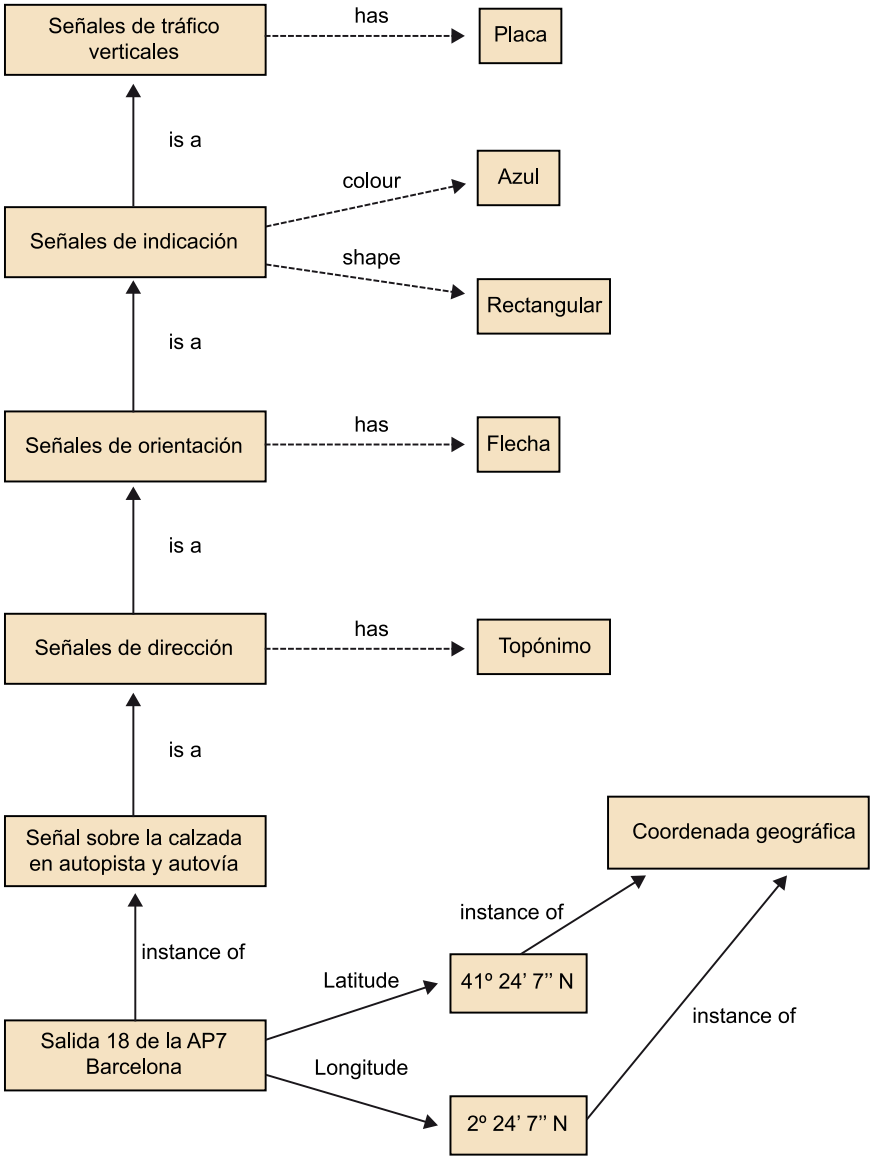
Otro elemento directamente afectado por el concepto de herencia son las instancias. Para indicar que un individuo pertenece a una clase, se utiliza un tipo de relación denominado *instancia de* (o *instance of* en inglés). Un ejemplo de instancia sería "La silla en la que estás sentado ahora mismo es una instancia de Silla". Al definir este conocimiento, el sistema podría inferir de manera automática que vuestra silla es un mueble.

Ejemplo 24. Uso de herencia en una representación

Circulando hacia Barcelona por la autopista AP7, veremos indicada su salida por medio de una señal vertical que contiene el texto "Salida 18, Barcelona". Estamos ante una instancia de una señal de dirección sobre la calzada de una autopista. Es una señal fácilmente identificable para los que conocen el código de circulación, puesto que es igual en forma al resto de las señales de la misma clase; es decir, tiene las mismas características.

De las clases a las que pertenece la señal "Salida 18, Barcelona" se puede inferir que esta es una placa rectangular y contiene una flecha.

Ilustración 3. Ejemplo de herencia en la especialización



Este esquema muestra cómo podemos representar todo este conocimiento de manera organizada para reflejar las entidades más generales y las específicas.

En la práctica, las fuentes de conocimiento declarativo están divididas en dos grandes bloques según el tipo de conocimiento que describen:

1) Conocimiento intensivo[6], cuando se describe el dominio por medio de conceptos generales (clases, superclases, relaciones y principios).

(6)En inglés, *intensive knowledge.*

(7)En inglés, *extensive knowledge.*

2) Conocimiento extensivo[7], cuando se describe el dominio mediante información concreta, como por ejemplo utilizando las instancias concretas propias del dominio.

Ejemplo 25. Definición de la clase *Planeta del Sistema Solar* de manera intensiva y extensiva

La clase *Planeta del Sistema Solar* podría definirse de manera intensiva o extensiva. De manera intensiva, la podríamos definir como un subtipo de *Planeta* con una restricción de integridad que diga que todas sus instancias tienen que orbitar alrededor del Sol. De manera extensiva, la clase se podría definir indicando que sus instancias son *Mercurio*, *Venus*, *La Tierra*, *Marte*, *Júpiter*, *Saturno*, *Urano*, *Neptuno* y *Plutón*.

1.3.7. Conocimiento lingüístico

Cuando hablamos de conocimiento lingüístico, lo primero que nos puede venir a la mente es un diccionario. El **diccionario** es una compilación de palabras de una lengua con sus definiciones. Sin embargo, un diccionario solo contiene la palabra principal, es decir, sin derivados (plurales, flexiones verbales o declinados).

Si a este recurso añadimos relaciones entre las palabras, relaciones léxicas que expresen similitud en el significado (sinonimia y antonimia), obtenemos un **tesauro**. Si no nos quedamos aquí y damos un paso más, y añadimos más relaciones léxicas (como la hiponimia, metonimia) y también agrupamos las palabras por conceptos, obtendríamos un recurso de conocimiento declarativo. Parte de este trabajo se ha llevado a cabo en diferentes proyectos, y WordNet es el más conocido.

Sinonimia, antonimia, hiponimia, metonimia

Sinonimia: relación semántica que se establece entre dos o más palabras diferentes que tienen el mismo significado.

Antonimia: relación semántica que se establece entre dos o más palabras de significados opuestos, es decir, entre palabras que significan lo contrario o lo inverso.

Hiponimia: relación semántica que se establece entre una unidad léxica (grupo de palabras con significado propio) y otra cuando esta se encuentra incluida dentro del campo léxico de la primera. Por ejemplo: *fruta* tiene los hipónimos *naranja*, *pera*, *manzana*. La relación vista a la inversa es la hiperonimia.

Metonimia: relación entre significados basada en una relación de contigüidad. Por ejemplo: "Se comió dos platos" (se comió el contenido del plato).

Ejemplo 26. WordNet

Si consultamos la palabra *car* del inglés en el recurso léxico del proyecto WordNet, obtendríamos el conjunto de conceptos del que forma parte. Si nos fijamos en el concepto principal, "vehículo de motor de cuatro ruedas, generalmente propulsado por un motor de combustión interna", vemos que incluye cuatro sinónimos o, dicho de otro modo, variantes de *car*: *auto*, *automobile*, *machine* y *motocar*.

Además, también podemos consultar cuál es su hiperónimo si su concepto es: "vehículo propulsado de manera autónoma que no va por vías". Este tiene dos variantes: *motor vehicle* y *automotive vehicle*. De este modo, podríamos ir navegando por la relación de hiperonimia hasta llegar al concepto raíz: *entity* (entidad).

El proyecto WordNet

El proyecto WordNet es un ejemplo de representación utilizada para el estudio del lenguaje. El proyecto reúne en una base de datos léxica nombres, verbos, adjetivos y adverbios del inglés en conjuntos de sinónimos, es decir, conceptos.

Además de este conocimiento que hemos mencionado, otros tipos de conocimiento como el conocimiento fonético, fonológico, morfológico, sintáctico, semántico e incluso el pragmático son conocimiento lingüístico. Sin embargo, no todo tiene que ser conocimiento declarativo, puesto que hay autores (como es el caso de Noam Chomsky) que lo ven como procedimental. Su postura se basa en que parte de este conocimiento es fruto de habilidades aprendidas y, por lo tanto, simulables como procedimientos.

Noam Chomsky

Noam Chomsky es uno de los científicos más importantes de la segunda mitad del siglo XX y un estudioso de la lingüística con derivaciones en las ciencias de la computación.

Podéis consultar más sobre su figura y su trabajo en la dirección siguiente:

The Noam Chomsky Website

Ejemplo 27. Fonología y fonética

La fonología es la ciencia que estudia los fonemas, es decir, la relación de los sonidos y las palabras. Con mucha frecuencia se confunde con la **fonética**, la ciencia que estudia el habla humana, y en concreto la producción de los sonidos. Las dos disciplinas tienen en común el hecho de que tratan aspectos del habla, pero desde puntos diferentes y complementarios.

El conocimiento fonológico es representado como conocimiento declarativo por medio de estructuras de árbol que definen las relaciones entre los fonemas y el conjunto de sílabas de una lengua. Estamos, pues, ante un conocimiento declarativo.

En cambio, en lo que respecta al conocimiento fonético, este será procedimental puesto que tiene que representar:

a) Procesos que transformen secuencias de fonemas en articulaciones del tracto vocal.

b) Procesos que conviertan vibraciones vocálicas en secuencias de fonemas.

2. La representación del conocimiento

La **representación del conocimiento** es un ámbito dentro del paradigma de la IA simbólica que estudia cómo especificar el conocimiento en un formato que soporte la resolución de problemas. Otro término muy utilizado en el ámbito de la representación del conocimiento es el de *ingeniería del conocimiento*. La ingeniería del conocimiento es la disciplina que estudia cómo integrar conocimiento en sistemas informáticos para resolver problemas complejos que, de lo contrario, necesitarían de intervención humana debido a su complejidad. Por lo tanto, podemos decir que la ingeniería del conocimiento no tan solo se ocupa de la representación del conocimiento, sino de cómo utilizarlo para solucionar un problema dado.

Los sistemas informáticos que utilizan conocimiento para llevar a cabo sus tareas se denominan sistemas basados en conocimiento (*knowledge-based systems*, en inglés). Quizá el tipo de sistema basado en conocimiento que mejor ejemplifica cómo se usa el conocimiento son los sistemas expertos. Los sistemas expertos utilizan la experiencia de un especialista del dominio (o de un conjunto de ellos) previamente formalizada para solucionar problemas complejos que requieren gran cantidad de conocimiento, ya sea teórico, práctico o tácito.

Ejemplo 28. Medicina y representación del conocimiento

La eficacia de un médico al resolver un caso médico no solo recae en el acierto de la diagnosis, sino que también es clave su habilidad en la elección del tratamiento más apropiado. En este proceso, lo ayudarán sus conocimientos de medicina y su experiencia en casos anteriores.

Si se representara la experiencia del médico en una base de conocimiento, se podría construir un sistema experto que emulase a un médico en las tareas de hacer diagnósticos, proponer tratamientos y justificar tanto los diagnósticos hechos como los tratamientos propuestos.

Si hasta ahora hemos estudiado el conocimiento (la información disponible sobre el mundo, el "qué" se sabe), ahora pasaremos a analizar su representación (la forma en que se codifica este conocimiento, el "cómo" se representa lo que se sabe). Esta cuestión no es banal, puesto que la manera de representar el conocimiento determinará cómo lo podremos manipular.

Cabe destacar que serán el problema y el dominio los que marcarán cómo representar el conocimiento en cada caso. Por otro lado, hay que subrayar que la eficiencia de la solución que proponemos vendrá condicionada, además de por la cantidad y el tipo de conocimiento que se trata, por lo bien que se codifica y se almacena.

2.1. Esquemas de representación del conocimiento

El principal objetivo de la representación del conocimiento es facilitar la extracción de conclusiones (inferencia) a partir del conocimiento, y para hacerlo es necesario expresar el conocimiento en una forma computable.

Un **esquema de representación** es un instrumento para transformar el conocimiento de un dominio en un **lenguaje simbólico** dotado de sintaxis y semántica para que pueda procesarse de manera computacional. Cada elemento de la representación (símbolo) codificará un fragmento de información perteneciente al dominio en cuestión.

La **sintaxis** describe las posibles formas de construir y combinar los elementos del lenguaje. Es decir, la sintaxis especifica cuál es el conjunto de sentencias de un lenguaje y qué combinaciones de sentencias son válidas.

La **semántica** determina el significado de los elementos del lenguaje y la relación entre estos y su referente en el dominio real. Es decir, la semántica determina de manera no ambigua qué significa cada una de las sentencias del lenguaje.

Es importante destacar que los esquemas de representación van más allá de las estructuras de datos o las bases de datos, las cuales se preocupan de manera exclusiva de cómo almacenar y recuperar datos de manera eficiente. Internamente, la implementación de un esquema de representación puede utilizar estructuras de datos y bases de datos, pero tiene que añadir otras estructuras y procesos que permitan hacer inferencia de conocimiento nuevo.

2.2. Propiedades de un esquema de representación

Un buen esquema de representación del conocimiento posibilita en todo momento un acceso rápido y preciso al contenido, así como las propiedades siguientes:

a) Representación apropiada (relativo a la representación). Debe tener la habilidad para representar todo el conocimiento que es necesario para el dominio en cuestión. Es decir, tiene que ser lo suficientemente rico como para contener los datos y la información mínima para dar una solución apropiada al problema.

b) Inferencia apropiada (relativo a la representación). Tiene que ser capaz de manipular las estructuras de la representación, de modo que en todo momento se puedan derivar nuevas estructuras asociadas con conocimiento nuevo inferido del antiguo.

c) Eficiencia inferencial (relativo al uso de la representación). Debe mejorar el proceso de inferencia mediante la inclusión de heurísticas y guías que lo agilicen, para, de este modo, optimizar el cómputo. Por lo tanto, tiene que representar aquellas características del problema que puedan ser explotadas de manera computacional.

d) Eficiencia de adquisición (relativo al uso de la representación). El esquema tiene que permitir la fácil incorporación de conocimiento nuevo.

Otras propiedades que conviene tener en cuenta son las siguientes.

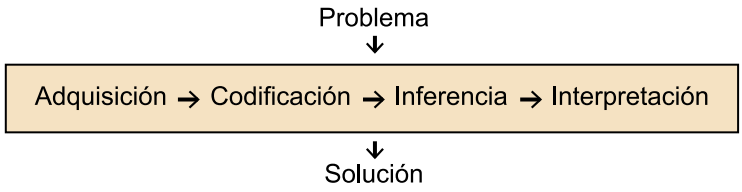
- **Claridad:** facilidad de identificar el conocimiento representado.

- **Naturalidad:** capacidad de representar el conocimiento en su forma original, de la manera más "natural" posible; es decir, el conocimiento no tiene que transformarse. La finalidad es que pequeños cambios en el problema requieran solo pequeños cambios en el esquema.

- **Modularidad:** posibilidad de fragmentar el conocimiento sin perder eficiencia ni eficacia.

2.3. Etapas de la representación del conocimiento

Dado un problema complejo que requiere el uso intensivo de conocimiento, podemos considerar las siguientes etapas en su resolución:



- El punto de partida es el **problema** que se quiere resolver, que marca cuál será el dominio de interés.

- La siguiente fase es la **adquisición** de conocimiento sobre el dominio de interés y el problema que se quiere resolver. Obtener el conocimiento de un dominio es una tarea muy compleja que involucra capacidades de procesado de la información como por ejemplo percepción, comunicación, asociación y razonamiento. Es vital obtener todo el conocimiento relevante para el problema y evitar conocimiento superfluo. De este modo, evitaremos problemas (errores, malas interpretaciones, redundancias, información sobrante, dispersión, etc.), minimizaremos los errores y mejoraremos el rendimiento en el acceso a la información.

Reflexión

En este módulo, dejaremos de lado la fase de adquisición por su complejidad y nos centraremos en la representación y utilización del conocimiento.

- Una vez obtenido el conocimiento del dominio, el paso siguiente es su **codificación** en un esquema apropiado. El tipo de codificación elegido y el modo como codificar la información condicionará hasta qué punto el conocimiento puede ser integrado con otras fuentes de información y compartido con terceros.

- Entonces, se aplica un motor de **inferencia** sobre el conocimiento guardado en el esquema de representación para extraer conclusiones sobre el problema.

- Para comunicar el resultado, el último paso es la interpretación de las conclusiones obtenidas en términos de hechos del dominio. De este modo, obtenemos una **solución** al problema planteado inicialmente.

Ejemplo 29. Etapas de la representación del conocimiento en la creación de un sistema experto médico

En el caso de que queramos crear un sistema experto para diagnosticar enfermedades, una posible distribución de las tareas por hacer en cada etapa es la siguiente:

1) Definición del problema: dar un diagnóstico a un paciente a partir de un conjunto de síntomas y saberlo justificar. Se puede limitar el sistema a una enfermedad o un conjunto de enfermedades.

2) Adquisición del conocimiento: reunir información sobre la relación entre síntomas y enfermedades. Hacer un conjunto de entrevistas a expertos en el dominio (médicos) para averiguar cómo pronostican enfermedades.

3) Codificación: se codificará el conocimiento obtenido en los pasos anteriores mediante reglas causales del tipo "si se dan el síntoma X, el síntoma Y y el síntoma Z entonces es probable que el paciente tenga la enfermedad E".

4) Inferencia: cuando un paciente informe al sistema de sus síntomas, el sistema experto usará un motor de inferencia para averiguar cuál de las posibles enfermedades es más probable que tenga el usuario y por qué (en caso de que esté enfermo, claro). En el caso de que el sistema necesite más información para llegar a alguna conclusión, la pedirá al paciente.

5) Interpretación: el sistema recogerá la información inferida en el apartado anterior y la presentará en un formato que sea comprensible para el paciente. En este caso, el sistema prepararía una respuesta en lenguaje natural para explicar al paciente su potencial enfermedad y un pequeño texto justificando este diagnóstico.

2.4. Retos de la representación del conocimiento

En las etapas que acabamos de identificar, aparecen varias cuestiones interrelacionadas sobre las que hay que tomar decisiones:

- ¿Cómo elegir cuál es el dominio relevante al problema? ¿Cómo acotar qué conocimiento queda dentro y cuál queda fuera?

- Una vez elegido el dominio de interés, ¿cuánto conocimiento se tiene que almacenar? ¿De qué tipo?

- ¿Cómo se adquiere este conocimiento del dominio?

- ¿Qué uso se hará de este conocimiento? Es decir, ¿qué tipo de inferencias se tienen que llevar a cabo para resolver el problema?

- A partir de todos los puntos anteriores, ¿cuál es el esquema de representación más adecuado para el problema?

Todas estas cuestiones nos sirven para darnos cuenta de lo complicado que será elegir una buena representación. Tenemos que hacer énfasis en que cuando hablamos de conocimiento, además de representarlo, no debemos olvidar en ningún momento que también deberemos procesarlo.

Además de estas decisiones, que son particulares para cada problema, hay una serie de retos más generales que aparecerán en cualquier problema de representación del conocimiento que consideremos: la imposibilidad de modelizarlo todo, el conocimiento de sentido común y el lenguaje natural. Antes de continuar, estudiaremos estos retos con más detalle.

2.4.1. El problema de modelizar el mundo

El éxito con el que un problema pueda ser solucionado dependerá directamente de nuestra habilidad para representar el conocimiento sobre el mundo relacionado con el problema. Aun así, no debemos olvidar que no trabajaremos sobre el mundo sino sobre una simplificación de este, que inevitablemente será imperfecta o incompleta.

Por supuesto, modelizar el mundo entero es una tarea de dimensiones astronómicas. Un volumen de información tan grande sería difícil de almacenar, y muy ineficiente su tratamiento computacional.

Por este motivo, generalmente diseñaremos estructuras y procedimientos para modelizar una parte de estos (un dominio concreto). De todos modos, aunque se trata de un objetivo terriblemente ambicioso, encontramos una variedad de proyectos que pretenden registrar gran cantidad de conocimiento del mundo en todos sus ámbitos. De entre estos, destacamos los proyectos Cyc, ConceptNet, WordNet (explicado anteriormente), The Knowledge Graph y Probase.

Cyc

Cyc es un proyecto privado iniciado en 1984 en el seno de una empresa de EE. UU. Desde entonces, el proyecto ha reunido millones de hechos, conceptos, relaciones entre estos, ideas, reglas, heurísticos, etcétera. Algunos ejemplos de su contenido son “Cada perro es un mamífero”, “Los perros mueren” o “Los animales de compañía tienden a ser apacibles”. Todo este conocimiento se ha codificado mediante un formalismo, el lenguaje denominado CycL (basado en el lenguaje Lisp). Hay que destacar la publicación en abierto de una parte del proyecto, con el nombre de OpenCyc.

El proyecto Cyc

Encontraréis más información sobre el proyecto Cyc en la página web de OpenCyc.

Ejemplo 30. Definición de información en Cyc

La regla "Los perros son mamíferos" se define en CycL como:

```
(#$genls #$Dogs #$Mammals)
```

Donde *Dogs* es la clase que representa a los perros; *Mammals*, la clase que representa a los mamíferos; y *genls* es la relación de generalización/especialización de CycL.

ConceptNet

ConceptNet es un proyecto que nace dentro del Massachusetts Institute of Technology (MIT), en el MIT Media Lab, con la colaboración de numerosos organismos y empresas de todo el mundo. El objetivo es describir el conocimiento básico, cultural y científico del mundo que las computadoras deben saber. El fruto de este trabajo es un inmenso grafo en el que los vértices representan conceptos y las aristas, las propiedades que los relacionan. Actualmente se encuentra en su versión 5 y se distribuye mediante la licencia Creative Commons.

El proyecto ConceptNet

Encontraréis más información sobre el proyecto ConceptNet en la página web de Conceptnet 5.

Ejemplo 31. Definición de información en ConceptNet

La regla básica "All dogs are mammals" se define en ConceptNet como:

```
dog --(belongs to the collection of)--> mammals
```

The Knowledge Graph

The Knowledge Graph es un proyecto de Google que tiene como objetivo crear una base de conocimiento para mejorar los resultados de su motor de búsqueda. Este conocimiento representará el mundo por medio de las entidades que lo forman: personas, lugares, películas, etc., así como las relaciones entre estas. Con este proyecto, Google quiere dar un paso adelante tecnológicamente, introduciendo conocimiento semántico dentro de la búsqueda. Este conocimiento semántico permite, entre otras cosas, tener información de contexto de los términos buscados por los usuarios y utilizar esta información para desambiguar búsquedas, refinarlas o ampliarlas.

El proyecto The Knowledge Graph

Encontraréis más información sobre el proyecto en la página web The Knowledge Graph; también en el blog de este proyecto.

Probase

Probase es el homónimo de The Knowledge Graph de Microsoft. Probase es una base de conocimiento que contiene conocimiento general y de sentido común, y tiene como objetivo dar apoyo a los programas informáticos en lo que respecta a entender mejor la comunicación humana. Su base de conocimiento, que contiene cerca de 3 millones de registros, se puebla automáticamente a partir de los millones de páginas web que hay en Internet y de los *logs* de búsqueda de los internautas.

El proyecto Probase

Encontraréis más información sobre el proyecto en la página web: Probase.

2.4.2. El problema del sentido común

Un chiste popular explica que el sentido común es "el menos común de los sentidos". Incluso a los seres humanos nos cuesta aplicar el sentido común en todas las situaciones, y podemos llegar a cometer errores evidentes.

Si queremos evitar que un sistema inteligente cometa este tipo de errores, tenemos que codificar el conocimiento de sentido común relativo al problema.

Veamos, como ejemplo, un texto de Marvin Minsky en el que habla sobre el sentido común y el razonamiento:

"Solo en matemáticas podemos decir (sentencias como) «si *a y b* son enteros, entonces $a + b = b + a$» . Consideremos ahora un hecho como «los pájaros pueden volar». Si pensamos que el razonamiento del sentido común es el mismo que el razonamiento lógico, entonces uno cree que hay principios generales de este tipo: «si Joe es un pájaro y los pájaros pueden volar, entonces Joe puede volar». ¿Consideramos que Joe es un avestruz o un pingüino? Bien, si es así podemos crear el axioma «si Joe es un pájaro y no es un avestruz ni un pingüino, entonces Joe puede volar». Sin embargo, ¿y si Joe está muerto? ¿O tiene los pies clavados?"

El razonamiento

"El razonamiento es la capacidad de los humanos de dar sentido a las cosas para establecer y verificar hechos, así como para cambiar o justificar prácticas y/o creencias."

Nikolas Kompridis (2000). *So we need something else for reason to mean* (págs. 271-295). [Traducción libre.]

Hay dos factores que deberemos tener en cuenta a la hora de intentar codificar el conocimiento del sentido común:

- Hay una gran cantidad de conocimiento de sentido común y no es factible representarlo todo. Habrá que elegir el conjunto mínimo suficiente para resolver el problema.

- Puede ser difícil identificar el conocimiento de sentido común, puesto que es tan evidente que acostumbra a darse por asumido. Por lo tanto, hay un riesgo elevado de pasar por alto este conocimiento en el esquema de representación. Será necesario hacer un esfuerzo adicional para explicitar este conocimiento durante la fase de adquisición.

Ejemplo 32. Uso del sentido común en problemas de posicionamiento

Los problemas de posicionamiento (*layout*) consisten en elegir la ubicación de un conjunto de objetos (puertas lógicas en circuito, maquinaria en una fábrica, etc.) en una superficie bidimensional acotada. El posicionamiento tiene que optimizar criterios como por ejemplo el tamaño o el rendimiento del sistema global.

Hay algunas restricciones de sentido común a la hora de posicionar los objetos, como por ejemplo el hecho de que dos objetos no se pueden solapar. Si nos olvidamos de representar este conocimiento, podemos obtener soluciones erróneas a nuestro problema. Por ejemplo, podríamos obtener como solución óptima "ubicar todos los objetos en la misma posición". Esta solución minimiza claramente la superficie ocupada y el tiempo de comunicación entre objetos, pero no se puede llevar a la práctica puesto que las leyes de la física impiden que dos objetos ocupen el mismo espacio de manera simultánea.

2.4.3. El problema del lenguaje natural

El lenguaje que utilizamos las personas como mecanismo para comunicarnos se denomina **lenguaje natural**. Este nombre se utiliza en contraposición a los conocidos como **lenguaje formales**, como por ejemplo los lenguajes de programación.

Lenguajes naturales y lenguajes formales

Lenguaje naturales: lenguaje producido de manera espontánea por los seres humanos, y cuyas reglas derivan del uso. Se utiliza en la comunicación humana.

Lenguaje formales: lenguajes artificiales definidos a partir de un conjunto de convenciones y reglas que permiten una comunicación precisa.

La ventaja del lenguaje natural es su gran **expresividad**: permite expresar cualquier cosa que se pueda representar simbólicamente y, por lo tanto, puede describir incluso elementos de arte, fotografías o emociones. Una segunda ventaja es la gran **cantidad de conocimiento** que está representado en lenguaje natural.

En consecuencia, gran parte de los primeros estudios efectuados en el campo de la representación del conocimiento estaban vinculados al lenguaje y la búsqueda de la información en el campo del lenguaje. Su base eran las investigaciones llevadas a cabo durante décadas en el análisis filosófico del lenguaje.

Por desgracia, la potencia del lenguaje natural genera un gran inconveniente: la **ambigüedad**. Una frase o palabra es ambigua cuando admite más de una interpretación posible. En estos casos, para elegir la interpretación correcta es necesario basarse en el contexto de la frase, que normalmente queda implícito.

Ejemplo 33. Ambigüedad del lenguaje natural

La frase siguiente:

"John vio a David en el bosque con los binóculos."

tiene dos interpretaciones posibles: la primera, que John vio a David por medio de los binóculos. No obstante, tenemos una segunda interpretación posible: John vio que David llevaba unos binóculos. Por lo tanto, nos falta conocer el contexto para saber a quién pertenecen los binóculos.

La ambigüedad es el motivo por el que los esquemas de representación se basan en lenguajes formales y no en el lenguaje natural. El uso del lenguaje natural se centrará mayoritariamente en la interacción persona-ordenador o bien en la fase de adquisición del conocimiento.

El **procesamiento del lenguaje natural** (*NLP*[8]) es el ámbito de la IA dedicado a la investigación sobre la comprensión y el tratamiento del lenguaje natural. Principalmente, las técnicas de procesamiento de lenguaje natural pertenecen a las disciplinas de aprendizaje automático y de extracción de datos[9], las que contribuyen al desarrollo de métodos de resolución de tareas como sumarización automática, búsqueda de respuestas[10], traducción automática, identificación de entidades[11], desambiguación lingüística, etc.

(8)Del inglés *natural language processing*.

(9)En inglés, *data mining*.

(10)En inglés, *question answering*.

(11)En inglés, *named-entity recognition*.

2.5. Los esquemas lógicos

Lectura complementaria

Para empezar la indagación en el campo del procesamiento del lenguaje natural, un buen libro de referencia es:

C. D. Manning; H. Schütze (1999). *Foundations of statistical natural language processing*. Cambridge: MIT Press.

La lógica matemática define muchos formalismos diferentes para codificar hechos y propiedades. Cada formalismo ofrece una cierta **capacidad expresiva** para representar el conocimiento en expresiones denominadas **fórmulas**. Además, cada formalismo establece unas **reglas de cálculo o inferencia** que permiten combinar fórmulas para extraer conclusiones a partir del conocimiento existente.

Indecidibles

Encontramos formalismos lógicos en los que el proceso de cálculo puede ser tan costoso que no hay garantía de su terminación: ¡el proceso podría quedar calculando de manera indefinida, sin llegar a dar ninguna respuesta!

El proceso de inferencia tiene un coste computacional que puede llegar a ser muy elevado y que no sea factible razonar sobre una base de conocimiento. Este coste está directamente vinculado a dos factores:

- El tamaño de la base de conocimiento: cuanto más conocimiento guardamos, más costoso es razonar sobre el mismo.
- La expresividad del formalismo lógico: cuanto más expresivo es el formalismo, las reglas de cálculo son más flexibles y, por lo tanto, hay muchas más maneras de aplicarlas.

La elección del formalismo lógico apropiado para un problema es necesariamente un compromiso. Hay que buscar un equilibrio entre la expresividad (para representar el conocimiento necesario) y la eficiencia del proceso de inferencia (para razonar con rapidez).

En el ámbito de la representación del conocimiento, un formalismo lógico que se utiliza frecuentemente es la **lógica de primer orden** (**LPO**), puesto que permite razonar tanto sobre objetos individuales como sobre conjuntos de objetos.

Uno de los componentes fundamentales de las fórmulas en LPO son los términos. Un **término** puede ser:

- Una **variable** (*x, y, z,* etc.), que puede tomar diferentes valores.
- Una **función** de 1 o más argumentos (*suma*(*x*,7), *padre*(*z*), *potencia*(*x*,2, *y*),...), en la que cada argumento puede ser una constante, variable u otra función.
- Una **constante**[12] (*peter, 2, blue*, etc.), que puede hacer referencia a un objeto del dominio o al valor de un atributo de un objeto.

(12)Las constantes se pueden ver como un caso particular de función con 0 argumentos.

Los términos se pueden relacionar entre sí definiendo predicados. Un **predicado** está formado por un símbolo (que indica el tipo de relación entre los términos) seguido de 0 o más términos. Por ejemplo, el predicado *Loves(john,*

mother(x)) establece la relación *Loves* entre los términos *john* y *mother(x)*. Este predicado, pues, nos podría servir para codificar que "John ama a la madre de *x*" (donde no sabemos quién es *x*).

La **interpretación** que hacemos de un predicado –es decir, el significado que damos a los símbolos de predicado, de función, las constantes, etc.– y la correspondencia entre este y el mundo real nos dirá si el predicado es **cierto (1)** o es **falso (0)**.

Los predicados se combinan en **fórmulas** mediante **conectivas** lógicas, como por ejemplo la conjunción ($\wedge$, también denominada *and* o *y-lógica*), la disyunción ($\vee$, también denominada *or* u *o-lógica*), la implicación ($\rightarrow$) y la negación ($\neg$). Un ejemplo de esto podría ser: "John ama a Lucy pero no a la hermana de Gema".

$$\texttt{Loves(john, lucy)} \wedge \neg \texttt{Loves(john, sister(gema))}$$

La interpretación de una fórmula se calcula a partir de la interpretación de cada una de sus subexpresiones, como se puede ver en la tabla siguiente.

Tabla 3. Tabla de verdad de conectivas lógicas: conjunción, disyunción, implicación y negación

A	**B**	**A $\wedge$ B**	**A $\vee$ B**	**A $\rightarrow$ B**	**$\neg$A**	**$\neg$B**
0	**0**	0	0	1	1	1
0	**1**	0	1	1	1	0
1	**0**	0	1	0	0	1
1	**1**	1	1	1	0	0

Por último, a las fórmulas también se pueden añadir **cuantificadores**, que especifican el valor que toman las variables que aparecen en las mismas. Hay dos tipos de cuantificadores:

- **Existencial** ($\exists$), que indica que la fórmula es cierta para algún valor de la variable. Por ejemplo, para indicar que "John es amado por alguien" utilizaríamos la fórmula siguiente:

$$\exists \texttt{x: Loves(x, john)}$$

- **Universal** ($\forall$), que indica que la fórmula es cierta para cualquier valor de la variable. Para decir que "John ama a todo el mundo que es amado por Peter":

$$\forall \texttt{x: Loves(peter, x)} \rightarrow \texttt{Loves(john, x)}$$

2.5.1. La implementación

Prolog es uno de los lenguajes más utilizados en la representación de esquemas lógicos. Desde su creación en 1972 por A. Colmerauer y P. Rousell, este lenguaje ha contribuido de manera sustancial a la evolución de la IA.

Un esquema Prolog consta de dos elementos: **hechos** (predicados que son ciertos) y **reglas** (patrones del tipo "si se cumplen un conjunto de condiciones, se puede afirmar este predicado"). El esquema responderá a cuestiones planteadas en forma de uno o más **objetivos**, predicados en los que alguno de sus argumentos puede ser variable. Prolog utilizará los hechos y las reglas para encontrar una asignación posible de valores a las variables que hacen cierto el predicado (respuesta "Yes"), o concluir que el predicado es falso para cualquier valor de las variables (respuesta "No"). Internamente, se utiliza un algoritmo de *backtracking* (vuelta atrás) para considerar todas las posibles combinaciones de valores de las variables.

Convenciones de Prolog

En Prolog, cualquier cadena que empieza por mayúscula se considera una variable: un símbolo que puede tomar cualquier valor. Los nombres de los predicados y las funciones tienen que empezar necesariamente por minúscula. Observad que la convención es diferente a lo habitual en lógica de primer orden, en la que los predicados se escriben en mayúscula y las funciones, constantes y variables, en minúscula.

Ejemplo 34. Definición de hechos utilizando Prolog

Consideremos el dominio de las relaciones familiares. Primero, definiremos un conjunto de hechos.

```
gender(sarah, female).
gender(peter, male).
gender(john, male).
parent(sarah, john).
parent(sarah, peter).
```

La parte derecha de una regla indica las condiciones que se tienen que cumplir y es una lista de predicados separada por comas. En la parte izquierda hay un único predicado: el que será cierto si se cumplen todas las condiciones de la parte derecha.

Ejemplo 35. Definición de predicados en Prolog y resolución de consultas

A continuación, definimos un conjunto de predicados para especificar qué es una persona, qué es una madre y qué es un hermano:

- si X es el progenitor de alguien, es una persona:

```
person(X) :- parent(X,_).
```

- si X es el progenitor de alguien y es una mujer, es su madre:

```
mother(X,Y) :- parent(X,Y), gender(X, female).
```

- X e Y son hermanos si tienen un progenitor Z común:

```
sibling(X,Y) :- parent(Z,X), parent(Z,Y).
```

Entonces, el sistema sería capaz de responder a las cuestiones siguientes:

```
person(sarah)           Respuesta: Yes
mother(X, peter)        Respuesta: Yes (X = sarah)
sibling(peter, sarah)   Respuesta: No
```

Una característica muy particular de Prolog es que los valores pueden ser listas de elementos: secuencias de elementos entre llaves y separadas por comas, como por ejemplo [ ] (lista vacía), [2] , [3, 17, 1, 3, 45]. El tratamiento de listas en Prolog se suele hacer de manera recursiva: por un lado, se trata el caso de la lista vacía; posteriormente se hace esto con el primer elemento de la lista, y se continúa tratando el resto de los elementos de manera recursiva.

Ejemplo 36. Tratamiento de listas en Prolog

Si definiésemos el siguiente predicado para indicar si alguno de los elementos de la lista del segundo parámetro es un ancestro del valor pasado en el primer parámetro, "Generar una lista ordenada de los ancestros femeninos de X":

```
ancestors(X, []) :- \+ mother(X, _).       (X has no mother)
ancestors(X, [ First | Rest ]) :-
   mother(First, X),   (Add X's mother as the first element)
   ancestors(First, Rest).              (Repeat recursively)
```

Entonces el sistema podría responder a las preguntas siguientes:

```
ancestors( sarah, X ).        Respuesta: Yes (X = [] )
ancestors( john, Y ).         Respuesta: Yes( Y = [sarah] )
ancestors( peter, [Y, Z] ).   Respuesta: No
```

2.6. Los esquemas basados en redes

A diferencia de la representación lógica, que se destina a la búsqueda del razonamiento, los esquemas basados en redes están principalmente destinados a formalizar el conocimiento que el hombre utiliza en su mundo. Por esta razón, este tipo de representación aparece de la mano de lingüistas y psicólogos en trabajos que tienen como finalidad la comprensión del lenguaje natural.

Otra apreciación respecto a la representación lógica es que esta pretende asignar valores verdaderos (valor que indica en qué medida una declaración es verdad) a las expresiones lógicas basadas en la interpretación del mundo. Esto provoca que la representación lógica no siempre sea suficiente para responder cuestiones como por ejemplo: "¿De qué está hecho el caucho?".

En cuanto a su forma, la representación en red sostiene el conocimiento por medio de una estructura de grafo, en la que los **vértices** representan objetos o conceptos del dominio del problema (por ejemplo, el "coche" o el color "rojo") y los **arcos** representan normalmente relaciones o asociaciones entre sí (por ejemplo, "el coche *es de color* rojo"). Tanto los arcos como los vértices pueden estar **anotados** con etiquetas que aportan información adicional.

Recordemos

Recordemos que un grafo etiquetado es un grafo que tiene asociada información a sus arcos (o aristas) mediante una etiqueta que contiene cualquier tipo de datos o información útil.

Los ejemplos más conocidos de esquemas basados en redes son las redes semánticas, los grafos conceptuales y las ontologías.

2.6.1. Las redes semánticas

Para entender qué son las redes semánticas[13], hay que hablar primero de lingüística. El análisis lingüístico está dividido mayoritariamente en cinco campos de estudio:

(13)Las redes semánticas constituyen una representación idónea para la visualización del conocimiento.

- El fonológico, en el que se estudia cómo se asigna el sonido a las palabras.
- El lexicológico, en el que se analiza cómo se categorizan las palabras según su morfología.
- El sintáctico, en el que se estudia cómo se valida el orden y la estructura de las palabras en una frase.
- El semántico, centrado en cómo se le asigna un significado a las palabras.
- El pragmático, en el que se estudia cómo se extrae la intención de uso de las palabras en un contexto concreto.

En este ámbito, las redes semánticas quieren proveer representaciones de conocimiento para el estudio de la semántica. La semántica intenta describir el significado de las palabras (permite desambiguar su sentido, en caso de ambigüedad) y las condiciones en las que los significados pueden interactuar para ser compatibles con los otros aspectos del lenguaje.

El significado de un concepto se define a partir de las relaciones hacia otros conceptos. De este modo, somos capaces de inferir conocimiento por medio de relaciones apropiadas como por ejemplo *is a* o *instance of*.

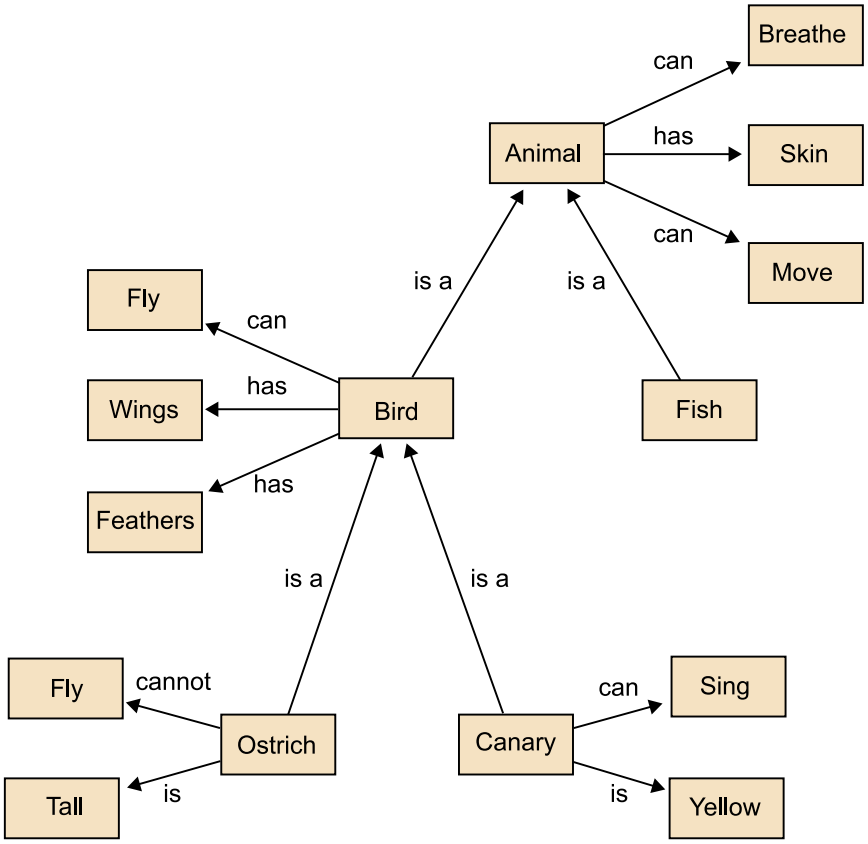
Por lo tanto, el conocimiento que mantienen las redes semánticas son definiciones de conceptos además de las relaciones entre ellas. Aparentemente, este es un patrón de estructura muy simple, pero ha sido utilizado con éxito para la construcción de muchas redes complejas, como es el caso del proyecto WordNet y su versión europea, EuroWordnet.

Muchas de estas redes han sido propuestas como modelos de representación de mapas conceptuales y mentales. Por el contrario, otros se han propuesto como componentes para la comprensión del lenguaje en sistemas de razonamiento.

Ejemplo 37. Red semántica con información sobre pájaros

El gráfico que acompaña este ejemplo (ilustración 4) es una muestra extraída de la red semántica desarrollada por A. Collins y R. Quillian (1969) en su investigación sobre el almacenamiento de información humana. Se trata de un buen ejemplo sobre cómo la jerarquía de la estructura tiene la capacidad de dar respuestas. Es decir, podemos hallar la respuesta a las preguntas siguientes: "¿Puede un canario cantar?"; "¿Es un canario un pájaro?"; "¿Puede un canario volar?"; o "¿Respira un canario?".

Ilustración 4. Red semántica



Estudios efectuados sobre cómo almacenan información los humanos revelan que utilizamos niveles de abstracción, y cada uno de estos contiene las propiedades que esperamos que sean generalizadas. Así pues, los seres humanos usamos de manera interna conocimiento heredable. Ante este conocimiento, la red semántica es un mecanismo idóneo para su representación.

Ved también

El mecanismo de generalización y especialización se explica con más detalle en el subapartado 1.3.6 de este módulo didáctico.

Ejemplo 38. Niveles de abstracción

La ilustración 4, además, nos ejemplifica una estructura con la repartición de la información en diferentes niveles de abstracción. En vez de almacenar de manera individual todas las características en el "canario", tenemos que la característica de "respirar" y la de "mover" están en el nivel de "animal". De este modo, en la estructura también podemos incluir el conocimiento de "pez", puesto que se trata de una especificación de "animal".

2.6.2. Los grafos conceptuales

Estructuralmente, un grafo conceptual es un grafo finito y conexo que tiene la propiedad de ser bipartito. Los vértices del grafo pueden ser conceptos o relaciones conceptuales. Este tipo de representación no asocia etiquetas a los arcos, sino que son los vértices de relaciones conceptuales los que representan las relaciones entre los conceptos. De este modo, los vértices de conceptos solo tendrán arcos hacia vértices de relaciones conceptuales, y también a la inversa. Por lo tanto, se garantiza la propiedad bipartita del grafo.

Grafos bipartitos y grafos conexos

Recordemos que un grafo $G = (V, A)$ es **bipartito** si existe una partición del conjunto de vértices, es decir, si se pueden separar en dos conjuntos disjuntos V_1 y V_2 de modo que las aristas solo conectan vértices de un conjunto con vértices del otro.

Un grafo $G = (V, A)$ es **conexo** si para cada par de vértices u y v de G hay un camino entre u y v.

Los vértices **concepto** representan objetos concretos o abstractos del mundo caracterizados por nuestra capacidad para formarnos una imagen mental de estos. Pueden incluir tanto conceptos generales –por ejemplo avión, teléfono u hotel– como instancias (objetos más específicos): el avión Boeing 747 con número de serie XW63783736E o el hotel Ritz de París.

Los vértices **relación conceptual** indican una relación que involucra uno o más conceptos. La ventaja de no utilizar etiquetas en los vértices es que esto simplifica la representación de relaciones con cualquier **aridad** (número de participantes en una relación). De este modo, una relación n-aria está representada por un nodo conceptual conectado con n vértices.

Un sistema basado en este tipo de representación estará formado por un conjunto de grafos en el que cada uno representa una única **proposición**. Las proposiciones representan parte de conocimiento del dominio.

Ejemplo 39. Grafo conceptual

La ilustración 5 representa un grafo conceptual sencillo de la proposición "un pájaro tiene plumas". En cambio, la ilustración 6 y la ilustración 7 añaden algo más de complejidad. En concreto, el grafo conceptual 2 representa "el pájaro puede volar con las alas". Observad que "volar" se relaciona conceptualmente con "pájaro", puesto que es el agente de la acción. También se relaciona con "alas", ya que estas son el objeto de vuelo. Por otro lado, el grafo conceptual 3 define que canario es un tipo de pájaro que es amarillo y canta.

Ilustración 5. Grafo conceptual 1

Ilustración 6. Grafo conceptual 2

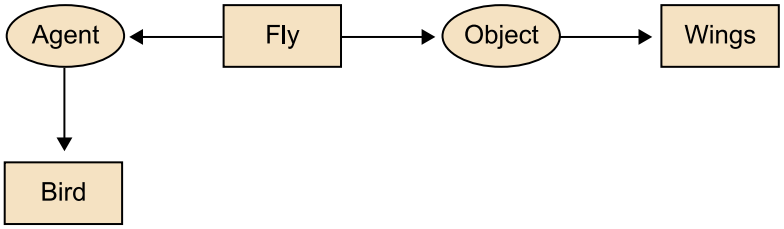
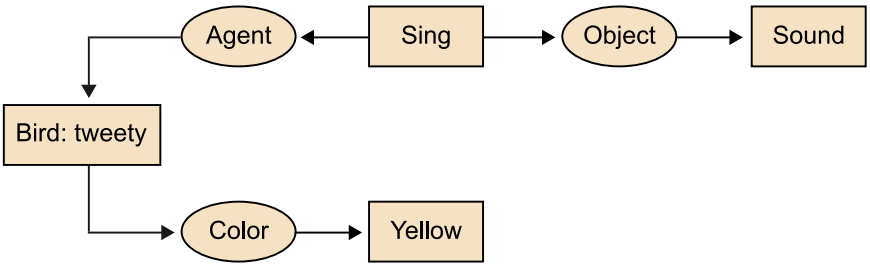
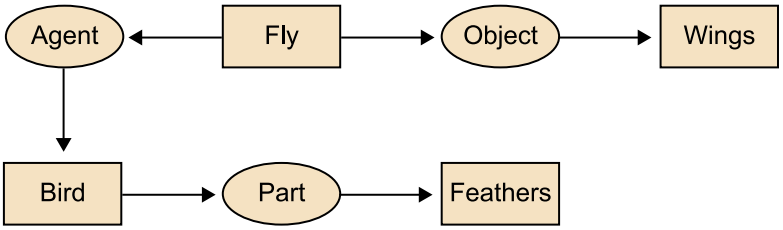


Ilustración 7. Grafo conceptual 3



La teoría asociada a los grafos conceptuales incluye un conjunto de operaciones para facilitar la creación de nuevos grafos a partir de los que ya hay. Las operaciones son la copia, la unión, la restricción y la simplificación. La ilustración 8 muestra la operación de unión entre los grafos conceptuales 1 y 2.

Ilustración 8. Ejemplo de unión de los grafo conceptuales 1 y 2

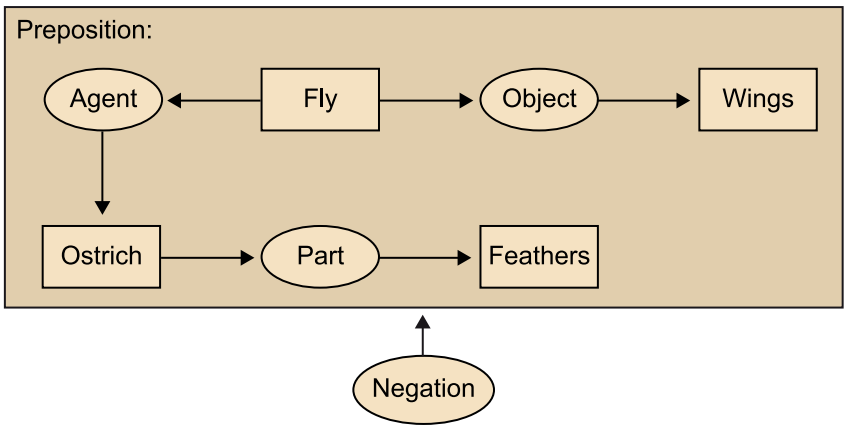


Los ejemplos vistos de grafos conceptuales demuestran lo sencillo que es representar objetos y sus características (en inglés, *conjunctive concepts*). En cambio, no hay ningún mecanismo estándar establecido para representar la negación o la disyunción. Algunos autores proponen utilizar la relación unaria "negation" asociada a una proposición.

Ejemplo 40. Operación de negación en grafos conceptuales

La ilustración 9 muestra la proposición "La avestruz no puede volar aunque tiene alas".

Ilustración 9. Operación de negación



2.6.3. La implementación

Los ejemplos de esquemas de representación que hemos mencionado en este subapartado (2.6) se utilizan de manera muy variada: para la recogida de conocimiento para la consulta humana, o bien para la consulta, la creación, la

modificación o la inferencia automática. Al mismo tiempo, también se aplican en muchos dominios como por ejemplo en el estudio del lenguaje (bases de conocimiento léxico), la web semántica o incluso la detección de plagio.

Esta variedad de usos es tan grande que ha provocado la proliferación de muchos lenguajes de programación destinados a la implementación de esquemas de representación. De estos lenguajes, hay que distinguir los que son de propósito **específico**, como es el caso del lenguaje CycL –creado por el proyecto Cyc (podéis ver el subapartado 1.3.1)–, de aquellos que son de propósito **general**, como el *Semantic Network Processing System* (*SNePS*) o el Lisp.

Para ilustrar cómo se utilizan estos lenguajes, veremos de qué manera implementar una red semántica en Lisp. La implementación es sencilla y, además, encontramos varios estilos para hacerlo.

Ejemplo 41. Red semántica implementada en Lisp

El siguiente código Lisp es la implementación del ejemplo 37 del subapartado 2.5.1 (red semántica sobre pájaros). Antes que nada, se define la función para crear relaciones de herencia entre conceptos (*entities*) de primer nivel (función *isa*), y a continuación para añadir atributos (función *attr*):

```
(defun isa (entity1 entity2)
   (setf (get entity1 'isa)
      (cons entity2 (get entity1 'isa))))

(defun attr (entity attribute value)
   (setf (get entity attribute)
      (cons value (get entity attribute))))
```

Ahora codificaremos el conocimiento propio del ejemplo utilizando las funciones definidas anteriormente.

```
(attr 'animal 'can 'breathe)
(attr 'animal 'can 'move)
(attr 'animal 'has 'skin)
(attr 'bird 'can 'fly)
(attr 'bird 'has 'wings)
(attr 'bird 'has 'feathers)
(isa 'bird 'animal)
(isa 'fish 'animal)
(attr 'canary 'can 'sing)
(attr 'canary 'is  'yellow)
(isa 'canary 'bird)
(attr 'ostrich 'cannot 'fly)
(attr 'ostrich 'is 'tall)
(isa 'ostrich 'bird)
```

Una vez tenemos todo el conocimiento creado y codificado, dotaremos a la representación de la funcionalidad para acceder al mismo y consultar los atributos de cada concepto, incluso aquellos que se heredan.

```
(defun get-attr (entity attr)
   (append (get entity attr)
      (get-from-parents (get entity 'isa) attr)
   )
)

(defun get-from-parents (parents attr)
   (cond ((null parents) nil)
      ((atom parents) (get-attr parents attr))
         (or (get-from-parents (car parents) attr)
            (get-from-parents (cdr parents) attr)
            )
```

```
            )
         )
      )

(defun entity-attr-value (entity attr value)
   (consp (member value (get-attr entity attr))
   )
)

(defun question-isa (entity parent)
   (consp (member parent (get-attr entity 'isa))
   )
)
```

Ahora ya estamos en disposición de llevar a cabo consultas:

- ¿Puede volar un canario?

```
(entity-attr-value 'canary 'can 'fly)
```

- ¿Tiene piel un pájaro?

```
(entity-attr-value 'bird 'has 'skin)
```

- ¿Un canario es un animal?

```
(question-isa 'canary 'animal)
```

También hay que destacar la proliferación de herramientas específicas para el tratamiento de ontologías, fruto del aumento en los últimos tiempos de su uso (en buena parte por la aplicación a la web semántica). Veréis más ejemplos relacionados con ontología y web semántica en el próximo módulo didáctico.

2.7. La representación estructurada

Aunque la representación estructurada se clasifica como otro tipo de esquema de representación, no deja de ser una variante de la representación en red. En concreto, la representación **estructurada** es un grafo ampliado en el que los vértices pasan a ser estructuras complejas con una colección de campos denominados *slots*.

Los ***slots*** permiten representar valores que pueden ir de un simple número hasta un procedimiento que lleva a cabo una tarea en particular. Los *slots* también pueden contener punteros, es decir, referencias a otra representación estructurada.

Los **marcos** y los ***scripts*** son ejemplos de esta representación.

2.7.1. Los marcos

Muchas veces, el conocimiento requiere ser organizado en unidades más complejas que simples vértices para que puedan representar situaciones u objetos muy conocidos del dominio. Estas unidades complejas de conocimiento se denominan marcos.

Marcos

Minsky (1975) describe un **marco** de la manera siguiente: "Cuando alguien se encuentra en una nueva situación (o cambia sustancialmente su perspectiva sobre un problema), selecciona desde su memoria una estructura denominada marco (*frame*). Un marco es un *framework* guardado en memoria que se puede adaptar para ajustarse a la realidad cambiando detalles según sea necesario".

Según Minsky, los humanos trabajamos con estructuras que recordamos de nuestra experiencia pasada y que ajustamos a las nuevas situaciones.

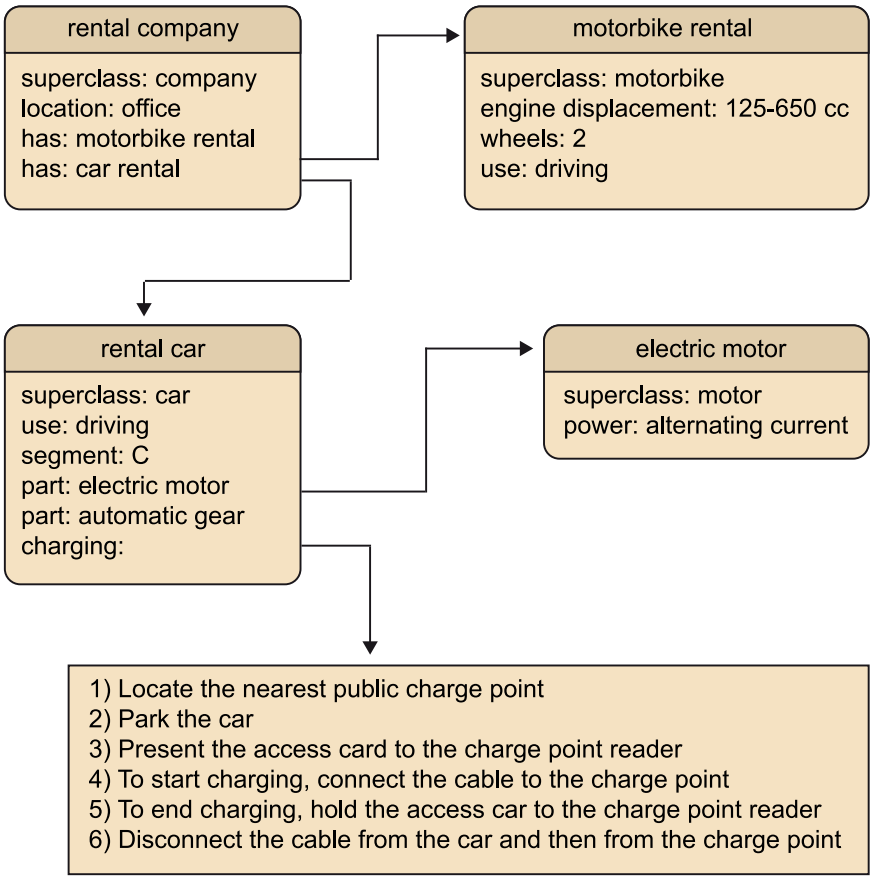
Ejemplo 42. Definición de un marco relacionado con alquiler de coches

Cualquier persona que tenga experiencia en conducir un coche posiblemente puede haber conducido más de un coche en su vida. Por este motivo, si alguna vez necesita alquilar uno, no tendrá ningún problema en saberlo conducir.

Cuando la empresa de alquiler de vehículos hace entrega del coche, el locatario espera encontrar unas puertas de acceso, un asiento para el conductor y para el resto de los pasajeros, un volante, unos pedales de gas y freno, etc. Pero además, percibe el color del coche, la posición del retrovisor y de los mandos de las luces, etc. También recordará las instrucciones de cómo llenar el depósito de gasolina: accionar la palanca debajo del volante, desenroscar el tapón exterior del depósito, etc. Todos estos detalles se organizan dentro de una estructura conceptual que representa un coche genérico.

La figura siguiente muestra cómo un coche de alquiler puede ser representado por un marco en el que se describen cada una de sus características. Vemos claro que el coche de alquiler es una especialización de la clase *Coche*, e incluye *slots* para detallar su uso, el segmento, las partes que lo componen y los pasos para cargarlo. Recordemos que los *slots* pueden contener valores (como son el uso, el segmento y el cambio), punteros a otros marcos (como es el caso del motor eléctrico) y también procedimientos, en este caso "cómo cargar de electricidad".

Ilustración 10. Marco que describe un coche de alquiler



Cada marco individual se puede ver como una estructura de datos que contiene la información relevante de sus características estereotipadas. En un marco, los *slots* contienen información como la siguiente:

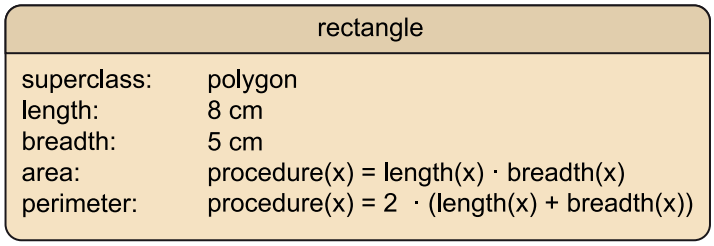
- Identificador del marco.
- Relaciones hacia otros marcos. En el ejemplo de este subapartado, el coche de alquiler es una instancia especial de *Coche* y la empresa de alquiler también alquila motos.
- Restricciones de integridad. Condiciones que determinan si un nuevo objeto encaja en el estereotipo definido por el marco (principios). Por ejemplo, la moto de alquiler tiene una cilindrada de entre 125 y 650 centímetros cúbicos.
- Información procedimental para el uso de las estructuras descritas. Es un mecanismo para incrementar la capacidad expresiva del marco.
- Información por defecto del marco. El valor de este *slot* se usa por defecto cuando no se informa de lo contrario. Por ejemplo, la moto tiene dos ruedas.

- Información de una instancia nueva. Serán valores que no están especificados hasta darse una instancia particular. Por ejemplo, el *Segmento* del coche no hay que especificarlo en la definición de *Coche*.

Ejemplo 43. *Slot* con información procedimental

En lugar de escribir un valor directamente en un *slot*, el gráfico siguiente nos muestra cómo los *slots área* y *perímetro* son procedimientos que permiten obtener un valor cuando se necesita. Con este mecanismo, podemos modificar la longitud del rectángulo y siempre obtendremos el área asociada al nuevo valor correctamente.

Ilustración 11. *Slots* con información procedimental



Para concluir, recordaremos que los marcos son útiles para simular el conocimiento de sentido común, generalmente conocido y estereotipado y que nos permite detallar todas las características típicas de lo que se modeliza.

La división explícita entre relaciones y atributos (*slots*) es una propiedad ventajosa que tiene el marco respecto a otros esquemas de representación. Hay que destacar también la posibilidad de descripción de su semántica a partir de su propia estructura.

Sin embargo, aun estando ante una representación que provee un método natural para representar entidades estereotipadas, clases, herencia y valores por defecto, G. F. Luger y W. A. Stubblefield avisan de los problemas todavía no resueltos que presentan los lenguajes de programación para su organización. Uno de estos problemas tiene relación con la herencia y se manifiesta cuando una instancia no puede heredar uno de sus *slots* de una superclase. En este caso, el usuario tiene que decir cómo se procede: bien eligiendo si hay que cancelar el *slot* en concreto o bien solo el valor que debería contener.

2.7.2. Los *scripts*

Scripts

G. F. Luger y W. A. Stubblefield (1998) definen el *script*[14] de esta manera: "Una representación estructurada que describe una secuencia estereotipada de eventos en un contexto particular".

(14)En español, 'guion'.

Lectura complementaria

El *script* fue diseñado originariamente por R. Schank y R. Abelson:

R. Schank; R. Abelson (1977). *Scripts, planes, goals and understanding: An inquiry into human knowledge structures*. Erlbaum.

Estamos, por lo tanto, frente a una estructura que tiene como objetivo la descripción de situaciones típicas por medio de la representación de secuencias de los eventos que suceden en ellas.

Su representación es una ampliación de los marcos, cuyos objetos se estructuran de modo que se pueda dar orden a las acciones que esperamos que pasen en aquella situación y los cambios de estado que se producen en ella.

Ejemplo 44. Uso de *scripts* para comer en un restaurante

R. Schank y R. Abelson proponen un ejemplo muy aclaratorio del uso de un *script*. Imaginemos la secuencia de escenarios posibles que tienen lugar cuando una persona come en un restaurante:

- Primero la persona entra dentro del restaurante.
- A continuación pide la comida al camarero.
- Después se la traen y se la come.
- Finalmente, se va del restaurante.

Como hemos visto en este ejemplo, el conjunto de **escenarios** sirve para ordenar y dividir la secuencia de eventos que tienen lugar en una situación, pero también es preciso descomponerlos con más detalle, en un ámbito de acciones y cambios de estado. Para hacerlo, se utilizará la teoría de **dependencias conceptuales**[15], enfocada a representar el conocimiento de los eventos que aparecen en las frases en lenguaje natural y facilitar la inferencia de conocimiento de manera independiente del lenguaje en el que se expresen.

(15)En inglés, *conceptual dependency*.

La representación de las dependencias conceptuales de una frase se construye con el significado que corresponde a las palabras que aparecen en la frase. Es decir, se separa el significado de las palabras. De este modo, tendremos que dos frases con el mismo significado dispondrán de una sola representación. Se utilizarán las **transiciones conceptuales** –formalismo para modelizar objetos y acciones del mundo– para representar de manera única el significado de las frases.

Lectura complementaria

Si queréis profundizar más en la teoría de las dependencias conceptuales, os recomendamos la lectura siguiente:

R. Schank (1969). *A conceptual dependency parser for natural language. Proceedings of the 1969 conference on computational linguistics*. Sång-Säby, Suecia (págs. 1-3).

Ejemplo 45. *Script* relacionado con una transición conceptual

Para describir la acción que un cliente hace en un restaurante para llegar a la mesa, hay una gran diversidad de frases posibles: "el cliente andará hacia la mesa", "el cliente se desplazará hasta la mesa", etc.

Todas las frases anteriores tienen el mismo significado. De este modo, para representarlas utilizaremos la transición conceptual PTRANS, destinada a las acciones de movimiento:

```
client PTRANS client to table
```

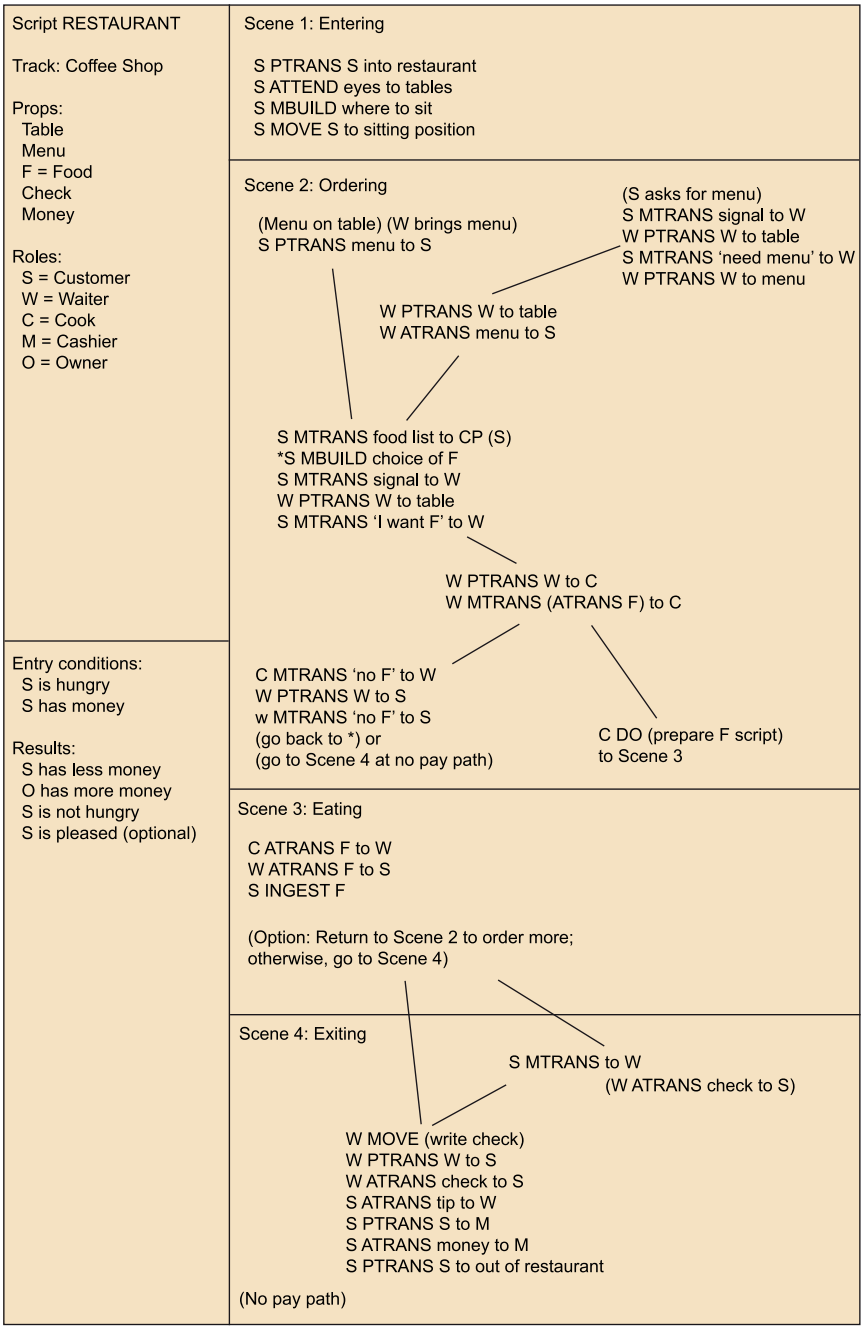
Además de PTRANS, también están MTRANS, utilizada para acciones de transferencia "dar" o "tomar"; y ATRANS, útil para las acciones que representan acciones mentales; etc.

Reflexión

Estamos ante un tipo de representación canónica del lenguaje natural que puede ser procesado por los ordenadores.

Ejemplo 46. Descripción de un *script* para un restaurante

El diagrama siguiente muestra un *script* que representa un restaurante con todas las acciones que suceden cuando un cliente entra a comer.



El *script* se compone de los elementos siguientes:

1) Las **condiciones de entrada**[16], que se tienen que satisfacer para que se pueda iniciar el *script*.

[16] En inglés, *entry conditions*.

Ejemplo 47. Condiciones de entrada en el restaurante

```
S is hungry
S has money
```

2) El **resultado**[17]; es un conjunto de condiciones que son ciertas cuando el *script* finaliza.

[17] En inglés, *results*.

Ejemplo 48. Resultados en el restaurante

```
S has less money
O has more money
S is not hungry
S is pleased (optional)
```

3) Las **propiedades**[18]; son los objetos involucrados en la situación que se describe.

(18)En inglés, abreviadas, *props*.

Ejemplo 49. Propiedades en el restaurante

```
Tables
Menu
F = Food
Check
Money
```

4) Los **roles**[19]; son las diferentes entidades o personas que intervienen en la situación descrita.

(19)En inglés, *roles*.

Ejemplo 50. Roles aplicables al dominio del restaurante

```
S = Customer
W = Waiter
C = Cook
M = Cashier
O = Owner
```

5) Los **escenarios**[20]; son las diferentes secuencias de eventos que se dan en el *script*.

(20)En inglés, *scenes*.

Ejemplo 51. Escenarios del restaurante

```
Scene 1: Entering
Scene 2: Ordering
Scene 3: Eating
Scene 4: Exiting
```

El *script* es un método de representación que está a caballo entre la representación estructurada y la procedimental (representación que describimos en el subapartado siguiente). Por este motivo, os avisamos de que en la bibliografía encontraréis algunos autores que lo clasifican de un modo u otro.

2.8. La representación procedimental

Las arquitecturas de este tipo codifican el "cómo llevar a cabo alguna tarea", y así modelizan el conocimiento procedimental, que ya hemos definido. En otras palabras, representan las habilidades que contiene el conocimiento, como por ejemplo conducir un coche, ir en bicicleta, etc. En definitiva, la representación procedimental incluye todo lo que suponga detallar los pasos para llevar a cabo una tarea.

Ved también

Podéis ver la definición de conocimiento procedimental que proponemos en esta asignatura en el subapartado 1.3.3 de este módulo didáctico.

Esta es una de las técnicas más utilizadas para representar conocimiento, puesto que tiene la ventaja de que no solo codifica los hechos, sino también la secuencia de operaciones. Sin embargo, esta ventaja es igualmente una debilidad por el hecho de tener entrelazados de manera intrínseca el conocimiento y su manipulación.

Podemos ver los lenguajes de programación como los mecanismos naturales para la codificación del conocimiento procedimental, además de los métodos basados en reglas.

Ejemplo 52. Representación procedimental de objetos gráficos

La infografía es uno de los campos en los que el diseño gráfico y la informática coinciden para trabajar juntos. La unión de las dos tecnologías ha producido un salto espectacular en el campo de la animación y el cine.

En este campo, trabajar con gráficos requiere no solo crear un conjunto estático de valores (como pueden ser todos los puntos que conforman una imagen en 3D), sino además los procedimientos que describen su animación.

Este tipo de representación permite la generación de formas y movimientos complejos. En concreto, se utiliza para modelizar fenómenos naturales: humo, plantas y colisión o rotación de objetos, entre otros.

Actor/Scriptor Animation System (*ASAS*) es un lenguaje de programación destinado a la creación de animaciones y gráficos por ordenador. Realmente es una ampliación de Lisp, pero provee una biblioteca de funcionalidades destinadas a la manipulación de objetos animados: vectores, polígonos, operaciones geométricas, etc.

El código siguiente ejemplifica cómo se define en *ASAS* la animación de la rotación de un cubo:

```
(defop spin-cube-actor
       (param: color)

       (actor (local: (angle 0)
                      (d-angle (quo 3 runtime))
                      (my-cube (recolor color cube)))
              (see (rotate angle y-axis my-cube))
              (define angle
                      (plus angle d-angle))))
```

Lectura complementaria

C. W. Reynolds (1982). "*Computer animation with scripts and actors*". *Computer graphics* (vol. 3, núm. 16).

2.8.1. Las reglas de producción

Las reglas de producción, también denominadas reglas *IF-THEN*, conforman una de las técnicas de representación del conocimiento más antiguas. Son un importante paradigma de representación aplicado en sistemas expertos: la planificación automática o la *action selection*. Además, su uso se ha popularizado en muchos dominios, como por ejemplo la medicina, como mecanismo útil para diagnosticar patologías médicas.

Las reglas representan el conocimiento utilizado en el formato SI-ENTONCES (*IF-THEN*), compuesto por dos partes esenciales:

- La parte del SI (*IF*), que indica el antecedente, la premisa, la condición o la situación.

- La parte del ENTONCES (*THEN*), que es el consecuente, la conclusión, la acción o la respuesta.

Ejemplo 53. Ejemplos de reglas SI-ENTONCES

SI conduces un vehículo **Y** se aproxima una ambulancia, **ENTONCES** tienes que reducir la velocidad **Y** debes ceder el paso a la ambulancia.

SI la temperatura corporal es superior **O** igual a 39 C, **ENTONCES** tiene fiebre.

Las ventajas que proporciona esta representación son las siguientes:

- Permite representar conocimiento declarativo y procedimental de manera conjunta.

- Es modular, puesto que cada una de las reglas es una porción de conocimiento independiente.

- Es fácil de manejar. Para modificar el conocimiento, solo hay que añadir o eliminar reglas.

- La forma SI-ENTONCES de la regla tiene con frecuencia una buena correlación con el lenguaje natural, con el objetivo de dar explicaciones.

- En algunos dominios, los sistemas basados en reglas consiguen algo parecido al modelo que el hombre utilizaría para solucionar por sí mismo el problema.

- Un sistema experto basado en reglas puede conseguir el mismo comportamiento que un experto humano en algunos dominios.

Por otro lado, también encontramos desventajas:

- El control de las reglas puede hacerse difuso, debido a que las reglas quizá tengan interacciones entre sí que pueden ser difíciles de controlar.

- El conjunto de reglas no tiene de manera intrínseca una estructura, lo que provoca que resulte difícil la manipulación de grandes conjuntos de reglas.

- No todos los métodos de resolución de problemas humanos se representan fácilmente en este formalismo.

- Estamos ante un método que, implementado en un sistema experto, no permite llevar a cabo una prueba rigurosa para la validación de su comportamiento.

2.8.2. Los programas

Los programas son una representación de unas secuencias de instrucciones, y cada una de estas puede ser ejecutada de manera automática para accionar nuevos eventos u obtener cambios de estado. Como sabéis, el orden secuencial de lectura del código no será el mismo que en la ejecución, ya que hay unos mecanismos especiales para indicar el control del flujo, y este control permite el salto de un punto a otro del código de manera independiente de su ejecución.

2.9. Las ontologías

La definición de ontologías más difundida en el ámbito de la informática es la que hizo Tomas Gruber en 1993, al definirlas como una especificación explícita y compartida de un dominio. En otros campos, como por ejemplo en el de la filosofía, el término *ontología* se utiliza desde la antigüedad (de hecho, desde que Aristóteles creó el concepto en la antigua Grecia). Ahora bien, para entender qué es una ontología, hay que analizar a fondo la definición de Gruber. Su definición habla de "conceptualización explícita, compartida y de un dominio". Estudiemos con calma qué significan los tres términos de esta expresión:

- "Conceptualización explícita" indica que una ontología es la representación de algo que se ha escrito de manera explícita en algún lugar; es decir, que no está solo en la mente de una persona. En este sentido, "explícita" también se refiere al hecho de que tiene que estar codificada en un formato que pueda ser codificado e interpretado por sistemas informáticos.

- "Compartida" significa que la representación de una ontología no puede ser la visión particular de alguien respecto a un dominio, sino que tiene que representar el conocimiento/información que una comunidad de personas tiene sobre un dominio.

- "De un dominio" indica que está enfocada a representar tan solo una parte de la realidad.

Una vez hemos clarificado estos conceptos, podemos definir el término de **ontología** de manera conceptual como la representación de parte de la realidad compartida por una comunidad y expresada en un esquema de representación que permite su interpretación por un sistema informático.

Aunque hoy día las ontologías acostumbran a usar esquemas de representación complejos basados en lógica descriptiva o en orientación a objetos, cualquier representación explícita y consensuada sobre un dominio es una onto-

logía. Así pues, una lista de conceptos de un dominio en un fichero de texto también podría ser una ontología, muy simple y con pocas posibilidades de inferencia, pero una ontología en efecto.

Según su poder expresivo, las ontologías se pueden clasificar en **ligeras** (que utilizan esquemas de representación simples y sin posibilidades de inferencia) o **pesadas** (con esquemas de representación complejos y altas capacidades de inferencia).

La representación ontológica refleja la conceptualización del dominio entre las personas de una comunidad, y es necesario un **consenso** para su creación. Por este motivo, una ontología describe la conceptualización en términos generales, en lugar de definir situaciones específicas.

Técnicamente, los elementos que constituyen una ontología son los conceptos, las relaciones, las instancias y los atributos:

- Los **conceptos** (vértice de la red) representan las categorías ontológicas que son relevantes en el dominio.

- Las **relaciones** (arcos de la red) conectan los conceptos de manera semántica.

- Las **instancias** (vértices hoja de la red) representan los objetos concretos del dominio, vértices que se clasifican por medio de las relaciones con los conceptos.

- Los **atributos** (combinación de vértices y arcos de la red) representan las propiedades y sus valores asignados de conceptos o instancias. Los valores son, al mismo tiempo, otros conceptos o instancias de la misma red.

Muchos autores, entre ellos T. R. Gruber (1993), comparten la opinión de que las ontologías tienen como propósito compartir y reutilizar el conocimiento que contienen, además de que han de tener un consenso (acuerdo) en el vocabulario utilizado.

Por este motivo, son muchas las fuentes de conocimiento del dominio médico representadas con ontologías. Un ejemplo es el de SNOMED-CT®, la ontología más grande, precisa y multilingüe de terminología clínica.

SNOMED CT

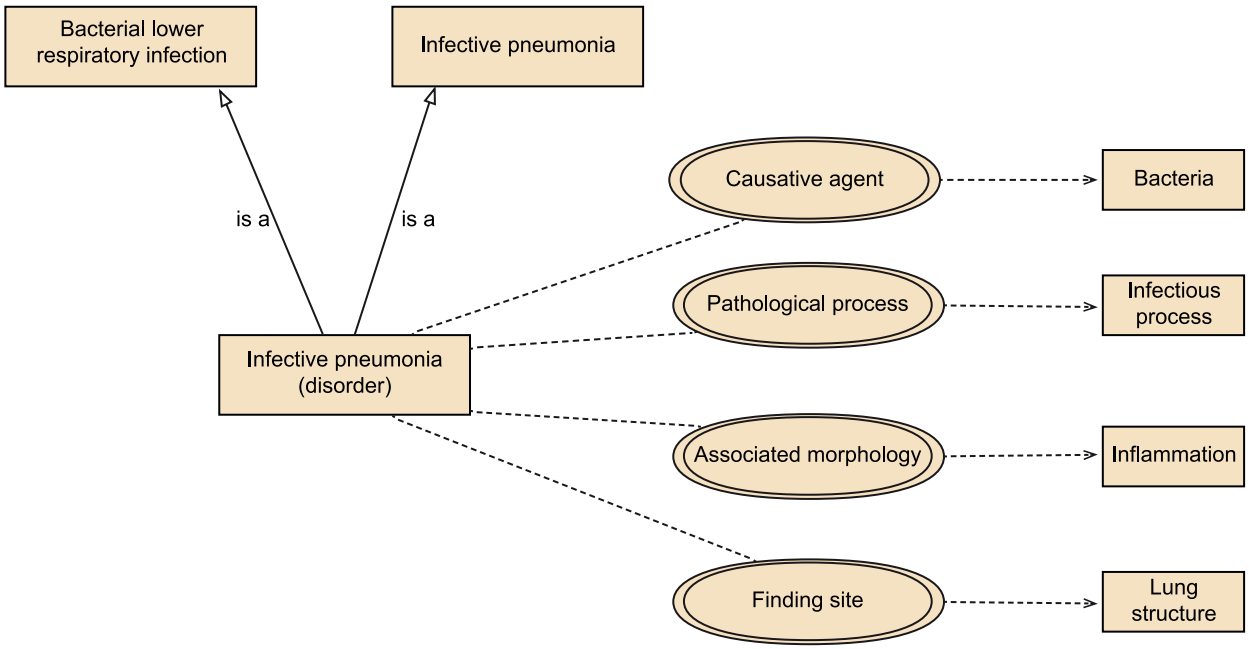
Para conocer más lo que es SNOMED CT, podéis ver el recurso de Internet siguiente:

SNOMED Clinical Terms® (SNOMED CT®) - OS National Library of Medicine

Ejemplo 54. Definición de un concepto SNOMED-CT® y sus relaciones y atributos

En la ilustración siguiente, mostramos el concepto *Infective pneumonia (disorder)* con todas sus relaciones y atributos.

Ilustración 12. Concepto *Infective pneumonia*



En cuanto a la visualización, una ontología se suele visualizar como una red semántica, como hemos hecho en este ejemplo.

Con mucha frecuencia, las ontologías se equiparan con jerarquías taxonómicas de clases. Sin embargo, recordemos que las taxonomías están limitadas a relacionar de padres a hijos o de subclase a superclase, y una ontología permite modelizar relaciones mucho más complejas.

2.10. Cuestiones para tener en cuenta

Son muchas las cuestiones que surgen cuando se utilizan algunos de los esquemas anteriormente descritos o tenemos la intención de representar algún tipo de conocimiento. Por ejemplo:

- ¿Qué relaciones son las más importantes y a la vez básicas?
- ¿Qué nivel de granularidad debería representar?
- ¿Cómo se tendrían que representar los conjuntos de los objetos?
- Dada una gran cantidad de conocimiento almacenado, ¿cómo se puede acceder a las partes relevantes?

A lo largo del módulo, se ha dado respuesta a algunas de estas cuestiones. A continuación, para dar luz alguna de las respuestas, reflexionamos sobre ciertos aspectos estructurales y funcionales.

Ved también

Recordemos que en el subapartado 1.3.6 se han explicado los tipos de relaciones de herencia, objetos y clases, bajo el epígrafe "La generalización y especialización del conocimiento". Además, en el subapartado 2.6.1 hemos entrado a diferenciar qué eran los atributos de los objetos y las relaciones.

2.10.1. Las relaciones

Existe una gran variedad de relaciones que conforman el conocimiento a partir de relacionar entidades, como ya hemos mencionado. Hasta ahora, hemos visto relaciones de herencia (*es un*, *tipo de*) y de instanciación (*instancia de*). Aparte de estas relaciones, normalmente denominadas taxonómicas, también hay otras denominadas relaciones no taxonómicas. Este segundo grupo incluye aquellas relaciones que no se utilizan para crear la taxonomía de herencia.

Otro tipo de relación muy común es la de meronimia, que incluye aquellas relaciones que permiten definir relaciones de pertenencia. Algunos ejemplos de esto son las relaciones *part of*, *has*, *member of*, *belongs to*, etc.

Ejemplo 55. Ejemplos de relaciones de meronimia

- Canadá pertenece a Norteamérica.
- Un motor es parte de un coche.
- Un árbol tiene hojas.
- Un dedo forma parte de la mano.

Otros tipos de relaciones utilizadas son las relaciones que indican propiedades semánticas de la lengua, como por ejemplo *sinónimo* y *antónimo*.

Ejemplo 56. Ejemplos de sinonimia

Alegre es sinónimo de *divertido*, *gracioso* y *risueño*.

Los tipos de relaciones que utilizaremos para representar el conocimiento en un problema concreto dependerán de distintos factores. Las relaciones de herencia y pertenencia tienen cabida en casi todas las representaciones. Por otro lado, las relaciones de instanciación tan solo son necesarias cuando la representación debe contener instancias o cuando queremos proveernos de ejemplos. Las relaciones que indican propiedades de la lengua serán muy útiles cuando se quiera interpretar u ofrecer información en lenguaje natural o cuando deseemos hacer tareas de alineación de conceptos (averiguar cuándo dos conceptos son el mismo o no); de lo contrario, pueden ser descartables.

Otro aspecto que conviene tener en cuenta a la hora de definir las relaciones es el esquema de representación. Según el esquema de representación que se utilice, algunas restricciones de integridad de las relaciones serán muy relevantes o totalmente irrelevantes. Por ejemplo, las restricciones que indican que una relación es simétrica, transitiva o funcional toman mucha relevancia si trabajamos con esquemas de representación basados en lógica descriptiva, mientras que las relaciones de cardinalidad pueden no ser tan relevantes. Además, no

será posible utilizar relaciones con una aridad superior a 2. Por otro lado, si utilizamos UML para representar el conocimiento, la aridad de las relaciones podrá ser superior a 2 y las restricciones de cardinalidad se hacen más relevantes.

2.10.2. Los atributos

Los atributos o las propiedades son relaciones binarias en las que cada uno de los participantes de la relación es un tipo de datos. Sorprendentemente, a día de hoy todavía encontramos muchas definiciones de tipos de datos y no parece haber un consenso claro sobre su semántica. En esta asignatura, para simplificar su identificación, estableceremos los tipos de datos según esta definición: un tipo de dato es una clase cuyos valores la identifican de manera unívoca. Por ejemplo, el número 3 es un tipo de dato porque dos números 3 son el mismo número. Lo mismo sucede con el texto "Juan" o con la fecha "3 de julio de 1974". No obstante, la clase *Libro* no es un tipo de datos porque podríamos tener dos libros con los mismos datos (ISBN, idioma, título, autor, etc.) pero que, aun así, se tratara de dos ejemplares físicos distintos.

Los atributos son muy importantes en la representación del conocimiento, porque nos permiten definir las características que tienen los elementos que modelizamos.

2.10.3. Restricciones de integridad

Tal y como se ha dicho, las restricciones de integridad son condiciones que tienen que satisfacer todas las instancias de nuestra representación. Hay muchas restricciones de integridad, de muchos tipos y que afectan a diferentes elementos (clases, objetos, relaciones, especializaciones, etc.).

Es importante definir las restricciones de integridad a la hora de representar el conocimiento, porque estas nos permitirán definir representaciones que tengan suficiente información como para detectar cuándo un problema dado (una posible instanciación) no pertenece al dominio representado (no es una instancia válida de la representación creada). Aparte de este aspecto práctico, las restricciones de integridad ofrecen conocimiento intensivo sobre el funcionamiento y las restricciones de los elementos del dominio.

Algunos ejemplos de restricciones de integridad

Ejemplos de restricciones de integridad serían "Todo estudiante de doctorado ha de tener un título universitario", "No puede haber dos restaurantes con el mismo nombre en la misma localidad", "La edad de una persona tiene que ser positiva", o bien "Una persona tan solo puede votar una vez en cada comicio electoral".

Las restricciones de integridad están soportadas de diferente manera en distintos lenguajes de representación de conocimiento. Cuanto más complejas sean las restricciones de integridad que definimos, más complejo (y normalmente más lento) será el motor de inferencia necesario para garantizar que se satisfacen.

2.10.4. La elección de la granularidad

El grado de detalle del conocimiento representado dependerá del problema y el tipo de conocimiento representado. Por este motivo, deberán tenerse en cuenta las consideraciones siguientes:

- Ser conscientes del volumen, tanto por los hechos generales (clases, conceptos) como por los más específicos (instancias, objetos).

- Las entidades más específicas suelen tener un mayor volumen y, por lo tanto, requieren más almacenamiento.

- Los hechos más generales no suelen ser los adecuados para la inferencia de conocimiento, puesto que tenemos menos especificación del dominio.

Ejemplo 57. Elección de la granularidad

Supongamos que estamos interesados en representar la información siguiente:

"Mary manchó a Peter."

La podemos representar como:

```
Manchar (agent(Mary), object(Peter))
```

Ahora, con esta representación es fácil responder cuestiones como esta:

"¿Quién manchó a Peter?"

Sin embargo, supongamos ahora que queremos saber:

"¿Vio Mary a Peter?"

Para responder a esta pregunta, necesitamos añadir algún hecho más; con los que tenemos, no podemos dar una respuesta. Elegimos generalizar la implicación de que *manchar* también significa *ver* lo que se mancha:

$$\texttt{Manchar(x,y)} \rightarrow \texttt{Ver(x,y)}$$

Ahora ya estamos en disposición de inferir una respuesta y darla.

2.10.5. Relaciones individuales y colectivas

Las clases pueden tener relaciones que las caracterizan y las definen, pero algunas forman parte de un conjunto más amplio. Es decir, hay afirmaciones ciertas de hechos a escala de conjunto, que no lo son a escala individual.

Ejemplo 58. Relaciones colectivas

Consideremos la afirmación siguiente:

"En Australia hay más ovejas que personas."

El hecho de que haya más ovejas que personas no es una propiedad individual de una oveja, sino que recae en el conjunto de ovejas. Por lo tanto, esta propiedad se tiene que definir para el conjunto de las ovejas.

Especificar una propiedad de manera colectiva en una entidad que representa un conjunto de objetos resulta mucho más eficiente que asociarla a cada uno de los elementos del conjunto que representa.

2.10.6. El reto del conocimiento incierto y heurístico

En el ámbito de las matemáticas, es posible enunciar propiedades universales y sentencias que tienen una validez absoluta, sin margen de duda o error. Desgraciadamente, fuera de este ámbito nuestro conocimiento está sujeto a factores que pueden introducir incertidumbre: errores de percepción, comunicación o interpretación; la subjetividad de los seres humanos; la aleatoriedad; etc.

Ejemplo 59. Ejemplos reales de conocimiento incierto

En el ámbito de la física, el resultado de cualquier experimento lleva asociado un margen de error, puesto que los aparatos de medida no tienen una precisión infinita.

En una prueba médica, siempre está presente el riesgo de obtener un resultado erróneo, ya sea por un error humano, una contaminación de la muestra, la interferencia de otra enfermedad, etc. Por este motivo, en ocasiones se pide repetir una prueba o hacer otras relacionadas para tener más fiabilidad en el diagnóstico. Un sistema inteligente que trabaje en el ámbito médico ha de tener en cuenta esta posibilidad de error a la hora de tomar decisiones.

El hecho de que un problema conlleva asociado un **conocimiento incierto** resulta más normal de lo que creemos, puesto que el conocimiento es casi siempre incompleto o incierto. Por este motivo, hay métodos de modelización especiales para indicar incertidumbre y que normalmente consisten en ponderar valores de confianza o pesos. La lógica difusa[21] es un ejemplo de esto.

[21] En inglés, *fuzzy logic*.

Ejemplo 60. Planteamiento de un problema con conocimiento difuso

Dados los conjuntos de ciudades siguientes:

X = {NYC, Paris} and Y = {Beijing, NYC, London}

¿Cómo podemos representar la idea de "muy lejos"? (J. Klir; T. A. Folger, 1988).

El **conocimiento heurístico** es fruto de la experimentación, y es el menos riguroso de los tipos de conocimiento. Un ejemplo de conocimiento heurístico sería la regla "Una persona no puede medir más de 10 metros". No hay ningún hecho que limite que una persona mida más de 10 metros, pero por lo que hemos observado hasta hoy día, damos esta suposición como cierta. Este tipo de conocimiento contiene una subjetividad que lo hace individual.

Algunos autores definen el **conocimiento heurístico** como "el conocimiento que subyace en el arte de adivinar bien".

Ejemplo 61. Procesamiento automático del lenguaje chino

En el procesamiento del lenguaje natural, una de las técnicas básicas para extraer información de un texto es llevar a cabo un análisis de las categorías gramaticales de cada una de las palabras. Es decir, detectar si la palabra es un nombre, un verbo, un adjetivo, un artículo, etc. Después de este primer análisis, se pueden llevar a cabo posteriormente procesamientos más complejos, como por ejemplo un análisis sintáctico.

Este análisis en un texto escrito en una lengua románica o germánica es relativamente más sencillo que en el caso del chino. Para poner un ejemplo, en inglés las palabras hacen uso de sufijos que indican la categoría gramatical: *translat**ion*** y *translat**e*** utilizan el sufijo *-ion* para el nombre y *- e* para el verbo. Desgraciadamente, en el chino las marcas flexivas que indican la categoría gramatical de las palabras son prácticamente inexistentes, lo que hace imposible distinguir si un mismo ideograma funciona como un nombre o como un verbo.

Para resolver este problema, se diseñan procedimientos que utilizan conocimiento heurístico basado en reglas que cuentan los patrones con ponderaciones para la clasificación gramatical de cada uno de los ideogramas en función del contexto (los otros ideogramas que lo acompañan en la oración).

Ejercicios de autoevaluación

1. Considerad el conocimiento implicado en los dominios o problemas siguientes y clasificad este conocimiento como tácito, explícito, declarativo, procedimental, heredable, etc., según corresponda. Tened en cuenta que puede haber más de un tipo de conocimiento implicado en cada dominio de conocimiento.

a) Un árbol genealógico de una familia.
b) La existencia de Dios.
c) Dibujar una casa sobre un papel en blanco.
d) Saber quiénes son los herederos de una persona.
e) Cocinar una tortilla de patatas.

2. Considerad el contexto de un agente de bolsa que compra y vende acciones. Identificad los ámbitos de conocimiento que están relacionados con este contexto, tal y como hemos hecho en el ejemplo 1.

3. Considerad la siguiente oración ambigua. Utilizando el esquema de grafo conceptual, representad sus diferentes significados posibles.

"John vio a David en el bosque con los binóculos."

4. Considerad la información siguiente y encontrad una red semántica que la represente:

a) El agua es líquida entre 0 y 100 grados.
b) El agua hierve a 100 grados.
c) El agua de la botella de Peter está congelada.
d) Picante es un tipo de agua.
e) Peter tiene agua del tipo picante en su botella.
f) Todos los líquidos tienen un punto de congelación.
g) El alcohol tiene una densidad de 0,79 gramos por centímetro cúbico.
h) El agua tiene una densidad de 1 gramo por centímetro cúbico.

5. Representad en la lógica de predicados la red semántica anterior con el lenguaje Prolog.

Solucionario

1. Describimos la clasificación para cada dominio o problema:

a) Es un conocimiento que se puede compartir porque representa hechos del mundo: relación de parentesco entre personas. Por lo tanto, lo clasificamos como conocimiento explícito y declarativo. Sin embargo, puesto que estas relaciones se pueden describir por medio de la lógica tradicional –el padre de vuestro padre es vuestro abuelo, y el padre del padre de vuestro padre es el abuelo de vuestro padre–, también se pueden subclasificar como conocimiento inferencial.

b) Lo clasificaremos como conocimiento tácito, debido a la inexistencia de hechos en el mundo real y a la dificultad de expresar creencias por parte de una persona.

c) El conocimiento para dibujar algo es una habilidad aprendida de la experiencia, y por lo tanto se trata de conocimiento tácito. Para dibujar sobre un papel, también hay que conocer un método y unos pasos, por lo que necesitamos igualmente el conocimiento procedimental. Sin embargo, hace falta además conocer las características de lo que se quiere dibujar (en este caso, una casa), por lo que también hay conocimiento declarativo.

d) Antes que nada, hay que saber qué normas rigen una herencia, por lo que se trata de un conocimiento que se puede comunicar (explícito). Estas normas se pueden ver como unas reglas o fórmulas (conocimiento procedimental). Por último, hay que conocer los hechos de parentesco de la persona cuyos herederos se quieren conocer. Este conocimiento sería declarativo.

e) Cocinar es un conocimiento que se aprende con la experiencia y, por lo tanto, es conocimiento tácito. Para cocinar una tortilla de patatas, hay que saber los ingredientes (conocimiento declarativo) y la receta, es decir, la secuencia de acciones necesarias para cocinar el plato (conocimiento procedimental).

2. Un agente de bolsa necesita el conocimiento siguiente para llevar a cabo la tarea de compraventa de valores.

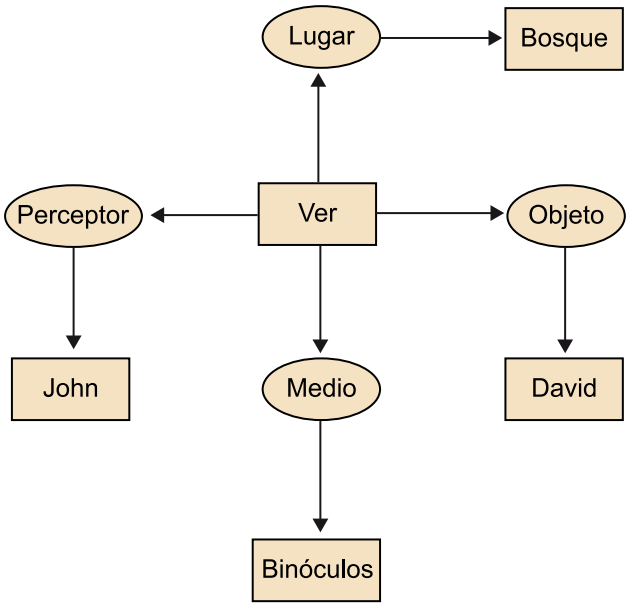
- Conocimiento sobre la economía:
  - Cálculo matemático y estadístico para el análisis de datos.
  - Administración de empresas, procedimientos administrativos.
  - Estados financieros, entender los distintos tipos de valores y el sistema de comercio.
  - Impuestos, leyes reguladoras y de protección y riesgos de inversiones.

- Conocimiento sobre sectores empresariales:
  - Sectores emergentes, tendencias y clasificación.
  - Hábitos industriales, farmacéuticos y tecnológicos.

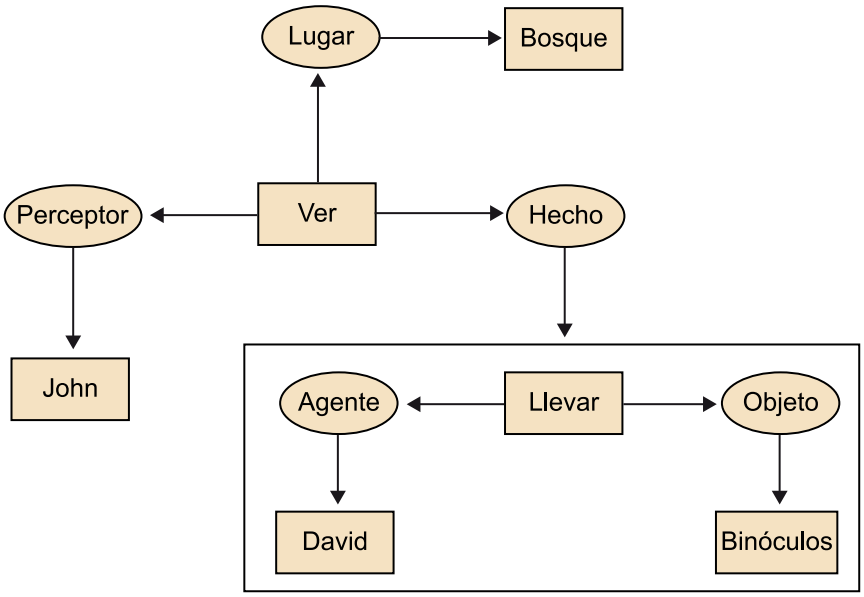
- Conocimiento sobre los mercados financieros:
  - Rentas variables nacionales o extranjeras, títulos, valores y acciones gubernamentales.
  - Confianza en los mercados de capital.
  - Relación entre la lógica y las emociones.

3. La oración presenta tres posibles interpretaciones, que exponemos a continuación.

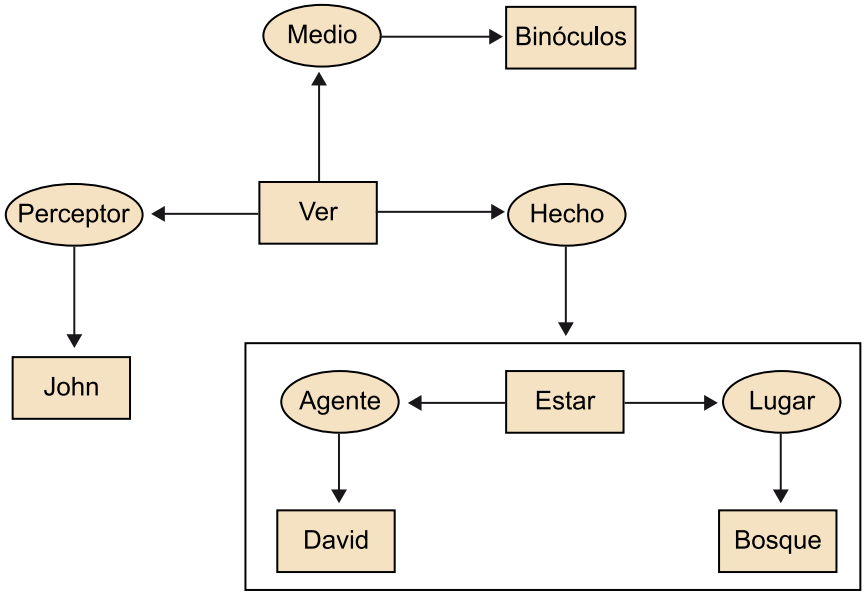
Interpretación 1: John estaba en el bosque y vio a David a través de unos binóculos.



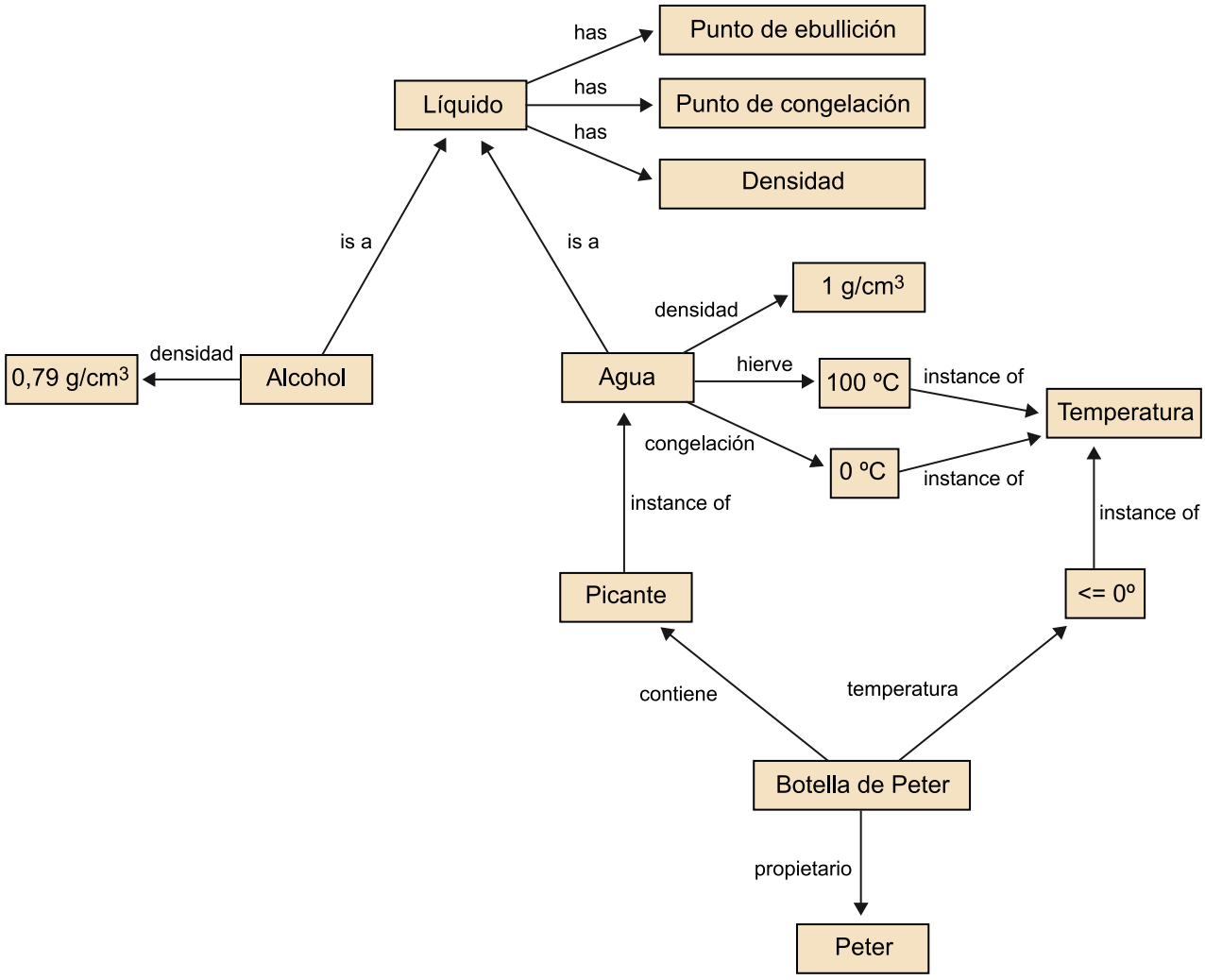
Interpretación 2: John estaba en el bosque y vio a David, que llevaba unos binóculos.



Interpretación 3: John, a través de unos binóculos, vio a David, que estaba en el bosque.



4. A continuación, os mostramos una posible red semántica que representa el conocimiento descrito. Debéis tener presente que puede haber otras combinaciones y variaciones de soluciones que son también correctas.



5. A continuación, os mostramos una posible representación de la red utilizando predicados de Prolog. Indicamos la traducción de cada apartado por separado para facilitar su comprensión. Una vez más, tened presente que hay muchas representaciones posibles:

a) El agua es líquida entre 0 y 100 grados.

```
compound(water).
compound(alcohol).
state(solid).
state(liquid).
state(gas).
freezingPoint(water, 0).
boilingPoint(water, 100).
inState(Object, liquid) :-
   madeFrom(Object, Material),
   temperature(Object, CurrentTemp),
   freezingPoint(Material, FPt),
   boilingPoint(Material, BPt),
   CurrentTemp > FPt,
   CurrentTemp < BPt.
```

Los hechos iniciales (que no serían necesarios) identifican los compuestos y los estados de la materia que se tendrán en cuenta. A continuación, definimos los puntos de congelación y de ebullición para el agua. Finalmente, definimos la regla que nos marca cuándo un objeto se encuentra en estado líquido: cuando la temperatura a la que se encuentra el objeto está entre el punto de congelación y ebullición del material que lo compone.

b) El agua hierve a 100 grados.

```
boilingPoint(water, 100).
inState(Object, gas) :-
   madeFrom(Object, Material),
   temperature(Object, CurrentTemp),
   boilingPoint(Material, BPt),
   CurrentTemp >= BPt.
```

El punto de ebullición del agua se ha definido en la sentencia anterior (se repite solo por claridad). A continuación, definimos la regla para concluir que un objeto se encuentra en estado gaseoso, muy similar a la regla para los estados líquidos.

c) El agua de la botella de Peter está congelada.

```
person(peter).
owner(bottle(Content), peter), madeFrom(Content, water), inState(Content, solid).
```

En este caso, no damos reglas sino que indicamos unos hechos. Utilizamos el predicado *owner* para describir al propietario de la botella y *bottle* para referirnos a la botella y su contenido. Observemos que todos los hechos referidos a la botella se enuncian de manera conjunta (separados por coma) para mantener el vínculo de la variable *Content* entre los tres hechos.

d) Picante es un tipo de agua.

```
compound(picante).
madeFrom(X, water) :- madeFrom(X, picante).
```

e) Peter tiene agua del tipo picante en su botella.

```
person(peter).
owner(bottle(X), peter), madeFrom(X, picante).
```

La regla es muy similar a la que hemos utilizado para la sentencia *c*.

f) Todos los líquidos tienen un punto de congelación.

```
freezingPoint(Material, _) :-
   madeFrom(Object, Material),
   inState(Object, liquid).
```

La regla explicita que si un objeto hecho de un cierto material se encuentra en estado líquido, entonces el material que lo compone tiene punto de congelación (no indicamos cuál es).

g. El alcohol tiene una densidad de 0,79 gramos por centímetro cúbico.

```
density(alcohol, 0.79).
```

Enunciamos esta información como un hecho de Prolog.

h. El agua tiene una densidad de 1 gramo por centímetro cúbico.

```
density(water, 1).
```

Nuevamente, utilizamos un hecho para codificar esta información.

Glosario

aridad *f* En los ámbitos de modelización conceptual y de representación del conocimiento, número de participantes en una relación. En el lenguaje matemático, la aridad es el número de argumentos u operandos de una función u operación.
Véase **relación**.

atributo *m* Característica de una instancia o categoría que puede tener un determinado valor. Se puede ver también como una relación entre una instancia (o categoría) y un tipo de datos.

categoría *f* Abstracción de las características que tienen en común un conjunto de instancias.
sin. **clase**

clase *f* Véase **categoría**.

concepto *m* Según el contexto, este término se puede utilizar para referirse a una idea abstracta (véase **instancia**) o bien a un conjunto de objetos.
Véase **categoría**.
sin. **entidad**

dominio *m* Área de conocimiento que engloba la información relevante para uno o más problemas relacionados.

entidad *f* Véase **concepto**.

esquema de representación *m* Instrumento para transformar el conocimiento de un dominio a un lenguaje simbólico que puede ser procesado de manera computacional.

frame *m* Véase **marco**.

grafo conceptual *m* Esquema conceptual en forma de grafo bipartito con dos tipos de nodos: concepto y relación conceptual. Las aristas indican qué conceptos participan en cada relación, lo que permite representar fácilmente relaciones con múltiples participantes.
Véase **red semántica**.

granularidad *f* Grado de detalle en el que se descompone un sistema.

guion *m* Esquema de representación que describe una secuencia estereotipada de eventos en un contexto particular.
en *script*

individuo *f* Véase **instancia**

instancia *f* Elemento concreto del mundo real, ya sea un objeto tangible o una idea abstracta.
sin. **individuo**
sin. **objeto**

instanciación *f* Acción de creación de una nueva instancia dentro de una categoría. Como parte de esta acción, se definen unos valores de sus atributos.

lenguaje formal *m* Lenguaje artificial definido a partir de un conjunto de convenciones y reglas que permiten una comunicación precisa. Se utiliza en el ámbito de las matemáticas o la lógica.

lenguaje natural *m* Lenguaje utilizado por los seres humanos para comunicarse entre sí, y cuyas reglas derivan del uso.

marco *m* Esquema de representación en forma de grafo ampliado en el que cada vértice puede tener asociada información declarativa o procedimental.
en frame

objeto *m* Véase **instancia**.

ontología *f* Esquema de representación que especifica de manera explícita y compartida la información de un dominio.

red semántica *f* Esquema de representación en forma de grafo etiquetado en el que los conceptos se representan como vértices del grafo y las relaciones, como aristas en las cuales

la etiqueta indica el tipo de relación. Así pues, este esquema está orientado a representar relaciones binarias.
Véase **grafo conceptual**.

relación *f* Vínculo entre dos o más instancias o categorías. Una relación se caracteriza por el tipo de vinculación, cuáles son los participantes, el papel de cada uno y las restricciones de integridad (participación obligatoria u opcional, número mínimo y máximo de participantes, etc.). Algunos ejemplos de tipos de vinculación son la generalización/especialización (*is a*) o la pertenencia (*part of*, *member of*, *belongs to*, etc.).

restricción de integridad *f* Condición que tienen que satisfacer obligatoriamente todas las instancias de una representación.

script *m* Véase **guion**.

semántica *f* Descripción del significado de los elementos de un lenguaje y la relación entre estos y su referente en un cierto dominio.

sintaxis *f* Conjunto de reglas que indican cómo se pueden construir y combinar los elementos de un lenguaje para formar sentencias.

slot *m* Denominación de los atributos en el contexto de los marcos.
Veáse **atributo**.

Bibliografía

Bibliografía básica

Davis, R.; Shrobe, H.; Szolovits, P. (1993). "What is knowledge representation?". *AI magazine* (vol. 1, núm. 14, págs. 17-33).

Kompridis, N. (2000). "So we need something else for reason to mean". *International journal of philosophical studies* (vol. 3, núm. 8, págs. 271-295).

Luger, G. L.; Stubblefield, W. (1998). *Artificial intelligence – Structures and strategies for complex problem solving* (3.ª ed.). Addison-Wesley.

Nonaka, I.; Takeuchi, H. (1995). *The knowledge creating company: how Japanese companies create the dynamics of innovation*. Nueva York: Oxford University Press.

Pole. D.; Mackworth, A. K.; Goebel, R. (1998). "Computational intelligence - A logical approach" (I-XVI, págs. 1-558). Oxford University Press.

Reynolds, C. W. (1982). "Computer animation with scripts and actors". *Computer graphics* (vol. 3, núm. 16).

Rich, E.; Knight, K. (2006). *Artificial intelligence* (págs.105-192). McGraw Hill.

Russell, S.; Norvig, P. (2010). *Artificial intelligence – A modern approach* (3.ª ed.). Person Education.

Bibliografía adicional

Rowley, J. (2007). "The wisdom hierarchy: representations of the DIKW hierarchy". *Journal of Information Science* (abril, vol. 33, núm. 2, pág 163-180).

Klir, G. J.; Folger, T. A. (1988). *Fuzzy sets, uncertainty and information* (1.ª ed.). Prentice Hall (enero). ISBN-13: 978-0133459845.

Sloman, A. (1979). "Epistemology and artificial intelligence". En: D. Michie (ed.). *Expert systems in the microelectronic age*. Edinburgh University Press.