Creación de dietas para ancianos: sistemas Basados en el conocimiento

IA FIB @ UPC

Carlos Bergillos, Roger Vilaseca, Adrià Cabeza

May 11, 2019

Contents

1 Introducción

Los **sistemas basados en el conocimiento (SBC)** son la parte de la Inteligencia Artificial donde se intenta hallar una solución utilizando el conocimiento. Hay ciertos problemas en la vida real que no se pueden resolver sin tener conocimiento específico sobre los elementos del dominio.

El motivo de esta práctica es comprender estos modelos de IA así que hemos atacado un problema con esta técnica: creación de menús para ancianos dependiendo de sus necesidades nutricionales y requisitos médicos.

Hemos desarrollado una ontología capaz de almacenar la información sobre los platos, los ingredientes, los nutrientes, las enfermedades y las limitaciones con **Protégé** y un sistema de reglas que describen el proceso de toma de decisiones usando **CLIPS**.

Para atacar el problema de una manera adecuada hemos utilizado las diferentes fases de la metodología de ingeniería del conocimiento:

- **Identificación**, donde identificaremos el problema y sus características y requisitos. Además también concretaremos el dominio para poder identificar las fuentes de conocimiento necesarias para su desarrollo.
- **Conceptualización**, donde describiremos en profundidad los conceptos que intervienen en el dominio del problema y la descomposición de este último en subproblemas.
- **Formalización**, donde partiendo de los conceptos de la fase anterior, procediremos a su formalización desarrollando una ontología. También, para la resolución del problema formalizaremos el conocimiento y una metodología de resolución.
- **Implementación**, donde dividiremos el problema en módulos para encapsular las diferentes partes del procedimiento en la resolución descrita en la fase anterior. Esta fase se basará en una metodología de desarrollo incremental, partiendo de un prototipo inicial que se va aumentando hasta conseguir un sistema final.
- **Prueba**, donde mediante juegos de prueba testearemos diferentes aspectos, en especial los más críticos, del sistema.

2 Identificación

En este primer apartado del trabajo, describiremos el problema, sus distintas partes y fuentes de infromación que requieren. Además añadiremos un análisis de viabilidad de utilizar un SBC para su resolución.

2.1 Descripción del problema

La alimentación es una pieza clave de nuestras vidas: nuestra manera de alimentarnos define nuestra salud. Con nuestra alimentación podemos prevenir el riesgo y agravamiento de enfermedades y mantener un nivel óptimo de salud. Es por eso que cuantos más años tenemos, más tenemos que vigilar nuestra alimentación.
En esta práctica desarrollaremos una solución para este problema. Una solución basada en un sistema basado en el conocimiento que sea capaz de generar menús semanales personalizados adecuados a las características de una persona según su edad, sus restricciones de dieta considearndo sus enfermedades y sus preferencias generales sobre la comida. Por ejemplo, si una persona es vegetariana y además diabética se lo generaría un horari sin carne y con poca cantidad de azúcares.

2.2 Viabilidad de utilizar un SBC

Si analizamos el problema podemos observar que no es un problema que se pueda resolver simplemente con algoritmos, sinó que el sistema requiere, para poder operar, de un conocimiento de nutrición salud.
Además nuestro objetivo, como ya hemos mencionado anteriormente, es dada la información de un anciano como input es dar una solución viable. Así que un SBC parece una opción bastante apropiada para enfretarse al problema.

Pese a que se podría utilizar otros métodos como la satisfacción de restricciones o incluso mediante una búsqueda en el espacio de soluciones creemos que la mejor manera de aprovechar el conocimiento implícito que requiere el problema es usando la potencia que nos proporciona un sistema basado en conocmiento.

2.3 Fuentes de conocimiento

Nuestras fuentes de conocimiento se pueden dividir en dos grandes pilares:

- **El usuario**: la persona que utilize nuestro sistema tendrá que indicar e introducir información acerca de su edad, sexo, enfermedades...para que podamos proporcionarle el mejor plan alimenticio.
- **Platos, ingredientes, enfermedades, nutrientes, etc**: el conjunto de conocimiento que necesita nuestro SBC para poder operar ha sido obtenido utilizando recursos en la red, especialmente se ha usado documentación especializada en el tema.

2.4 Objetivos y resultados esperados del sistema

El objetivo final del problema es mostrar un menú para toda la semana que sea adecuado a las necesidades alimentícias del usuario.

Para esto el sistema tiene que cumplir una serie de objetivos:

- Obtener todos los datos del usuario que sean necesarios para poder utilizar el conocimiento del sistema y así poder devolver el mejor resultado posible
- Inferir sobre las características del usuario de forma que obtengamos un conjunto de platos específicos para el usuario
- Presentar al usuario el menú obtenido por el sistema

Así pues esperamos de nuesto sistema un comportamiento que cumpla todos los objetivos indicados arriba para finalmente obtener unos resultados aceptables que esten concorde con el conocimiento del sistema.

3 Conceptualización

En este apartado describiremos todos los conceptos del dominio.

3.1 Conceptos del dominio

- **Persona**: las características necesarias para definir correctamente a la persona (usuario) son:
  - **Edad**: tiene que ser >= 65 ya que el sistema está orientado a ancianos

  - **Género**: ya que dependiendo del sexo se tienen que seguir unas restricciones alimentícias u otras.
  - **Nivel de ejercicio semanal**: que puede ser COMPLETAR
  - **Enfermedades que padece**
  - **Preferencias alimenticias**: que pueden ser Vegetariano, Vegano, COMPLETAR
- **Platos**
  - **Temporadas**: pueden ser Otoño, Verano, Invierno y/o Primavera
  - **Ingredientes**
- **Ingredientes**: las caracerístias necesarias para definir correctamente a un ingrediente son las siguientes:
  - **Nutrientes** Los nutrientes que contiene el ingrediente en cuestión
  - **Nombre**
  - **Temporadas**: pueden ser Otoño, Verano, Invierno y/o Primavera
  - **Tipo**:
    * **FICAR**
    * **ELS**
    * **DIFERENTS TIPUS D'INGREDIENTS QUE TENIM**
- **Nutrientes**: formado por los valores nutricionales que hemos considerado más importantes y adecuados para realizar nuestro sistema. Entre ellos se encuentran:
  - **Grasa**
  - **Carbohidratos**
  - **Calorías**
  - **Sucrosa**
  - **Glucosa**
  - **Fructosa**
  - **Lactosa**
  - **Cafeína**
  - **Alcohol**
  - **Azúcar**
  - **Colesterol**
  - **Grasas de las cuáles saturadas**
  - **Vitamina B5**
  - **Vitamina B6**
  - **Vitamina B12**
- **Enfermedades**: cada enfermedad, además de su nombre la cual nos sirve para definirla en sí, tenemos para cada enfermedad el conocimiento de qué limitaciones alimentícias presentan.
  - **Nombre**
  - **Limitaciones**: pueden ser limitaciones asociadas a nutrientes o a ingredientes.

3.2 Problemas y subproblemas

3.3 Ejemplos del conocimiento extraído del dominio

3.4 Flujo de razonamiento

4 Formalización

4.1 Desarrollo de la ontología

Para desarrollar correctamente la ontología hemos procurado que todos los conceptos del dominio sean representables. En este caso, los términos más importantes de la ontología son los **platos**, puesto que nuestra solución será un conjunto de ellos, los **ingredientes** y sus **características nutricionales** y los conceptos relacionados con el usuario: **preferencias** y **enfermedades**. Además para poder representar bien los susodichos conceptos, hemos creído conveniente hacer uso de clases auxiliares para definir cantidades. Así pues, por ejemplo para definir que un plato contiene una cantidad de un ingrediente en concreto usaríamos la clase *IngredientQuantity*.

4.1.1 Definición de clases y jerarquía

La ontología que hemos realizado para nuestro SBC contiene diversas clases. Se puede observar en la figura

Figure 1: Ontología de nuestro SBC

class_menu

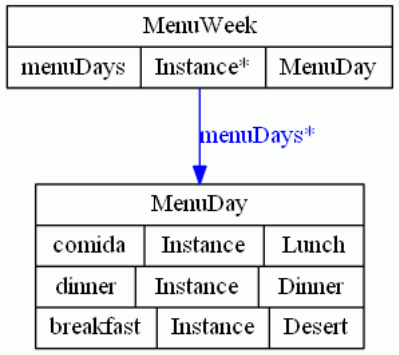

Figure 2: Clase Menu

Definimos toda la información de un menú en esta clase. Tenemos la cantidad de días que contiene nuestro plan alimenticio y disitintas instancias de menús diarios los cuales contienen información acerca de sus comidas mediante a sus instancias desayuno, comida y cena.

Los atributos son:

- **menuDays**: menú diario
  - comida: comida de un menú diario
  - dinner: cena de un menú diario
  - breakfast: desayuno de un menú diario

class_Course

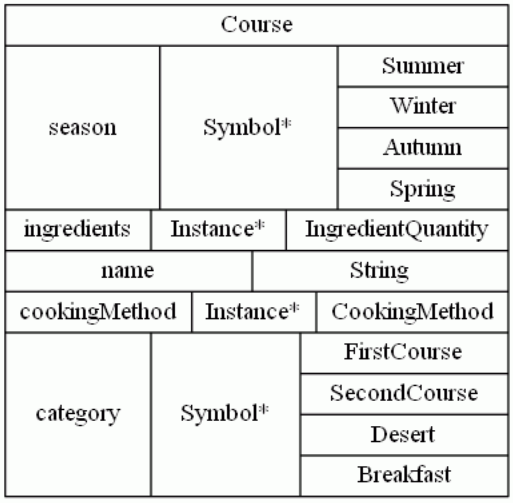

Figure 3: Clase Platos

Definimos aquí toda la información que contiene un plato. Tenemos información acerca de qué estación pertenece el plato, los ingredientes que contiene y sus cantidades además de la categoría a la que pertenece que puede ser primer plato, segundo plato, postre o desayuno,

Los atributos son:

- **season**: estación del año; puede ser Otoño, Verano, Invierno o Primavera
- **ingredientes**: ingredientes que contiene un plato y su cantidad
- **name**: nombre del plato
- **category**: categoría del plato; puede ser primer plato, segundo plato, desayuno o postres.

class_Ingredient

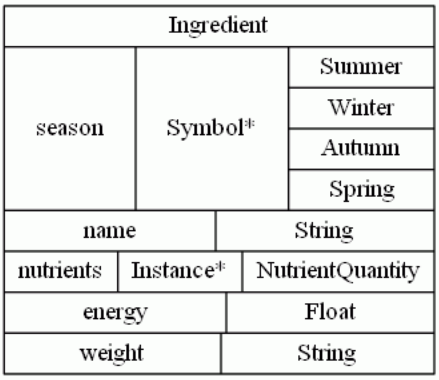

Figure 4: Clase Ingredientes

Definimos aquí toda la información que contiene un ingrediente. Tenemos de manera parecida a los platos, una estación asociada al ingrediente. Por ejemplo: las castañas estarían asociadas al otoño. Además también disponemos de los valores nutricionales que presenta el ingrediente.

Los atributos son:

- **season**: estación del año; puede ser Otoño, Verano, Invierno o Primavera
- **name**: nombre del ingrediente
- **nutrients**: nutrientes del ingrediente

class_IngredientQuantity

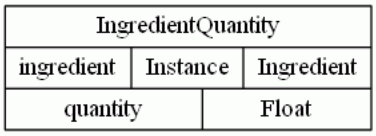

Figure 5: Clase Enfermedad

Para poder expresar la cantidad de ingredientes que contiene un plato hacemos uso de esta clase la cual contiene una referencia a una instancia ingrediente y la cantidad que hay en ese plato del ingrediente.

Los atributos son:

- **ingredient**: ingrediente
- **quantity**: cantidad del ingrediente

Definimos en esta clase toda la información que contiene un nutriente... TODO FINISH IT TODO continuar amb els atributs

class_NutrientQuantity

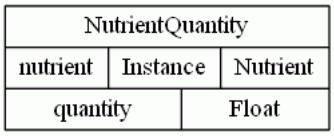

Figure 6: Clase Cantidad de Nutrientes

Para poder expresar la cantidad de nutrientes que contiene un ingrediente en concreto hacemos uso de esta clase. Esta dispone de una referencia a una instancia de nutriente además de la cantidad que ese nutriente en ese ingrediente (las unidades han sido normalizadas en gramos).

Los atributos son:

- **nutrient**: nutriente
- **quantity**: cantidad del nutriente

class_Disease

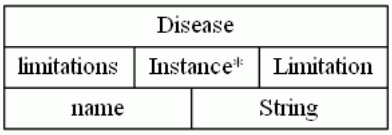

Figure 7: Clase Enfermedad

Definimos en esta clase toda la información que contiene una enfermedad. En nuestro caso, una enfermedad presenta una serie de limitaciones alimentícias que pueden estar referidas a un ingrediente o a un tipo de nutriente.

Los atributos son:

- **limitations**: limitaciones alimentícias que suponen la enfermedad
- **name**: nombre de la enfermedad

class_Limitation

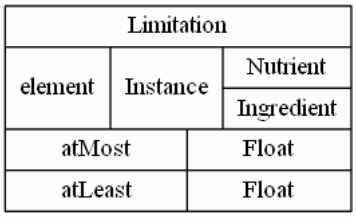

Figure 8: Clase Limitación

Para ser capaces de expresar una limitación o incluso una preferencia alimentícia, hemos hecho uso de esta clase para poder relacionar un nutriente o un ingrediente con una limitación de cantidad

AtMost(como máximo) o atLeast(como mínimo).

Los atributos son:

- **element**: elemento sobre el que se aplica la limitación, puede ser un nutriente o un ingrediente
- **atMost**: valor que representa el tope máximo necesario para cumplir la limitación
- **atLeast**: valor que representa el mínimo necesario para cumplir la limitación

4.2 Método de resolución

explicar problemas que intervienen en la resolución

5 Implementación

5.1 Creación de la ontología

Para generar la ontología hemos usado el programa Protégé añadiendo las diferentes características anteriormente mencionadas en los apartados
En cuanto a las instancias, hay una distinción básica entre ellas: estáticas y dinámicas. Hay algunas que se han creado dinámicamente durante la ejecucción del programa como podrían ser Persona o MenúDía. Para las estáticas en cambio (Nutriente, Plato, Ingrediente, IngredienteCantidad, NutrienteCantidad, Enfermedad o Limitaciones) hemos usado diversas bases de datos y un script en python para crearlas a priori y tener los archivos .pins necesarios.

5.2 Módulos

THIS GOTTA BE WRITTEN STILL

5.3 Prototipos

Tal y como se ha mencionado anteriormente, hemos seguido para el desarrollo de la práctica un método incremental por prototipos. Este tipo de desarrollo nos ha permitido avanzar de manera más organizada y paulatina.
En un inicio, desarrollemos nuestro **prototipo inicial**: una ontología bastante completa (muy parecida a la que se ha utilizado en la versión final) y un sistema que hacía preguntas y creaba un menú totalmente random. Nos sirvió para empezar a familiarizarnos con **Protégé** y **CLIPS** y en cómo tenían que ser las instancias. Después iteremos sobre ese prototipo: creemos un script en **Python** para facilitarnos la creación de las instancias y empezemos a añadir las primeras reglas básicas (i.e. restricciones de vegetariano).
Entonces después de hacer unos pequeños cambios en la ontología y por consiguiente en el script para crear instancias, procedimos a crear, de manera progresiva, las diferentes reglas inherentes al conocimiento.

6 Juego de Prueba

Una vez implementado nuestro SBC, hemos realizado una serie de pruebas enfocadas a testear diferentes puntos de la aplicación. Así pues, mediante estas pruebas, hemos podido comprovar el correcto funcionamento del sistema, tanto a nivel general como a nivel particular.

6.1 Prueba 1:

6.2 Prueba 2:

6.3 Prueba 3:

6.4 Prueba 4:

6.5 Prueba 5:

6.6 Prueba 6:

7 Conclusión

Podemos concluir que hemos aprendido sobre SBC y sobre el desarrollo de éstos pues hemos sido capaces de desarrollar un SBC functional basado en otologías y reglas de producción.

No obstante, cabe mencionar que debido a la complejidad inherente a la representación del conocimiento y el poco tiempo del que se ha dispuesto para realizar la práctica, el sistema podría ser mejorado con más conocimiento y más instancias.

A pesar de eso, hemos profundizado en el subcampo de los SBC y hemos conseguido realizar uno con utilidad práctica, que cumple los objetivos y resultados esperados.