Creación de dietas para ancianos: sistemas Basados en el conocimiento

IA FIB @ UPC

Carlos Bergillos, Roger Vilaseca, Adrià Cabeza

20 de mayo de 2019

Índice

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

Uno de los mejores métodos para trabajar con grandes problemas como este es dividirlos en varios subproblemas más pequeños para poderlo solucionar.

En nuestro caso lo vamos a dividir entre los siguientes 4:

3.2.1. Recogida de información personal

En esta parte del problema realizaremos varias preguntas al usuario (descritas en el apartado anterior), donde obtendremos la suficiente información para luego poder ser usado y obtener un menú adecuado a las condiciones y gustos de la persona.

3.2.2. Análisis de los datos personales

Una vez obtenidos los datos de la persona, el experto clasificará mediante una puntuación los diferentes platos según lo adecuados que sean para la persona. En esta parte también vamos a descartar aquellos platos que la persona no pueda tener en su menú en ninguna circunstancia.

3.2.3. Filtrado de platos para el menú

Vamos a coger los diferentes platos y los vamos a repartir mediante ... FALTA ACABAR UN COP DECIDIM L?ALGORITME FINAL

3.2.4. Impresión del menú

Una vez tengamos el menú construido vamos a sacar por pantalla el menú final de la semana, donde por cada plato habrá sus ingredientes. Después del menú va a salir la información nutricional del menú obtenido, comparados con los resultados que deberíamos tener a partir de la información que nos ha dado la persona en el primer subproblema.

3.3. Ejemplos del conocimiento extraído del dominio

A partir de información que obtenemos del dominio podemos llegar a clasificar los platos en función de las características y peticiones de cada persona.

Por ejemplo a partir de las siguientes características o preferencias que podríamos obtener a partir de las preguntas hechas, obtendríamos los siguientes resultados.

- En el caso de los vegetarianos vamos a quitar de los posibles platos aquellos que tengan algún ingrediente del tipos carne.
- Si la persona tuviera diabetes vamos a intentar que los platos sugeridos tengan poca cantidades de azúcares.
- En cambio si la persona responde que le gusta un alimento en concreto vamos a puntuar de una forma más alta aquellos platos que contengan el alimento en cuestión.

3.4. Flujo de razonamiento

Para poder clasificar correctamente los platos en nuestro problema a partir de las preguntas hechas.

Primero de todo pasaremos un filtro donde se eliminaran del conjunto de platos, aquellos que la persona no puede tomar en ninguna circunstancia.

A continuación los platos que hayan pasado de la opción anterior serán puntuados para prioriza que platos debe o no debe tomar dicha persona.

Al final obtendremos una lista ordenada de los mejores platos para la persona, con los cuales diseñaremos el menú más adecuado para cada usuario.

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 puedens observar en la Figura 1.

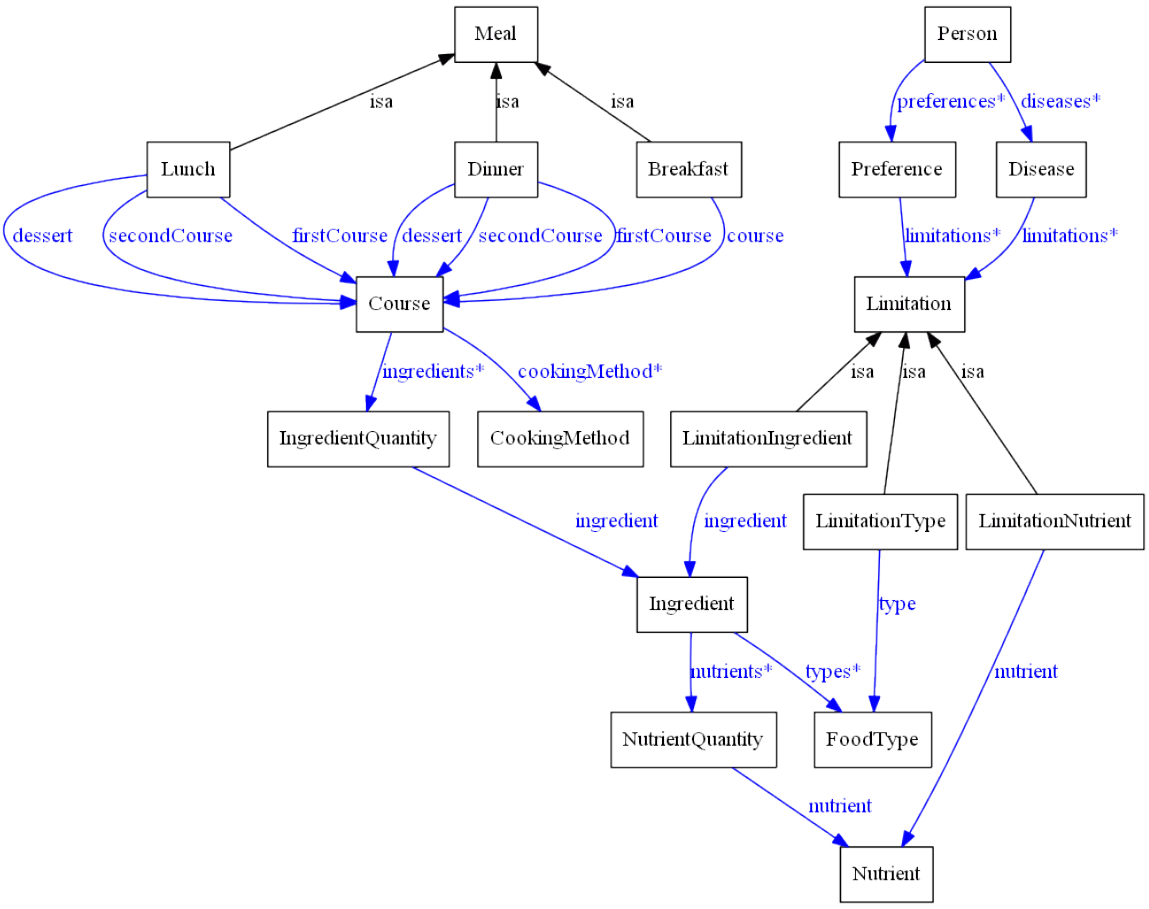

Figura 1: Ontología completa de nuestro SBC

A continuación detallaremos cada una de las clases.

Clase *Course*

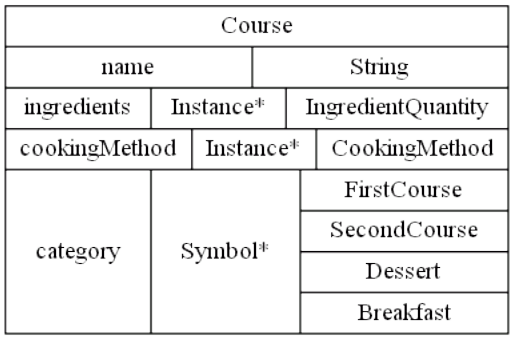

Figura 2: Clase *Course*

Definimos aquí toda la información que contiene un plato. Tenemos, entre otras cosas, la información acerca de 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:

- **name**: nombre del plato
- **ingredients**: lista de ingredientes que contiene el plato, y sus cantidades
- **cookingMethod**: método que se utiliza para cocinar el plato; i.e. hervir, freír...
- **category**: categoría del plato; puede ser primer plato, segundo plato, desayuno o postre.

Clase *Ingredient*

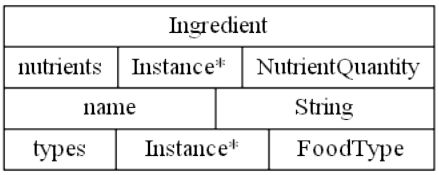

Figura 3: Clase *Ingredient*

Definimos aquí toda la información que contiene un ingrediente. Por ejemplo, los valores nutricionales que presenta el ingrediente.

Los atributos son:

- **name**: nombre del ingrediente

- **nutrients**: lista de nutrientes del ingrediente, y sus cantidades

Clase *IngredientQuantity*

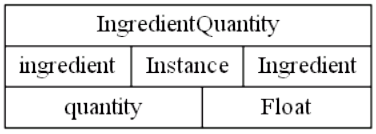

Figura 4: Clase *IngredientQuantity*

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

Clase *Nutrient*

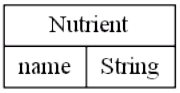

Figura 5: Clase *Nutrient*

Definimos en esta clase toda la información que contiene un nutriente. Aunque la clase actualmente solo contiente un atributo para el nombre, representa un concepto lo suficientemente importante en nuestra ontología como para merecer su propia clase, ya que en futuros proyectos esta clase podria requerir otros atributos.

Los atributos son:

- **name**: nombre del nutriente

Clase *NutrientQuantity*

NutrientQuantity		
nutrient	Instance	Nutrient
quantity		Float

Figura 6: Clase *NutrientQuantity*

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

Clase *Disease*

Disease		
limitations	Instance*	Limitation
name		String

Figura 7: Clase *Disease*

Definimos en esta clase toda la información que contiene una enfermedad. En nuestro caso, una enfermedad presenta una serie de limitaciones alimentícias.

Los atributos son:

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

Clase *Preference*

Preference		
limitations	Instance*	Limitation
name		String

Figura 8: Clase *Preference*

Definimos en esta clase toda la información que contiene una preferencia. En la práctica, una instancia de esta clase se comporta de forma muy parecida a la de una de la clase *Disease*. Pero

aunque las preferencias tengan tambien una lista de limitaciones, estas suelen ser menos estrictas que las de una enfermedad, e incluso puede tratarse de preferencias positivas, en las que se quiere propiciar el uso de ciertos alimentos o nutrientes.

Los atributos son:

- **limitations**: limitaciones positivas o negativas asociadas a la preferencia
- **name**: nombre de la preferencia

Clase *Limitation*

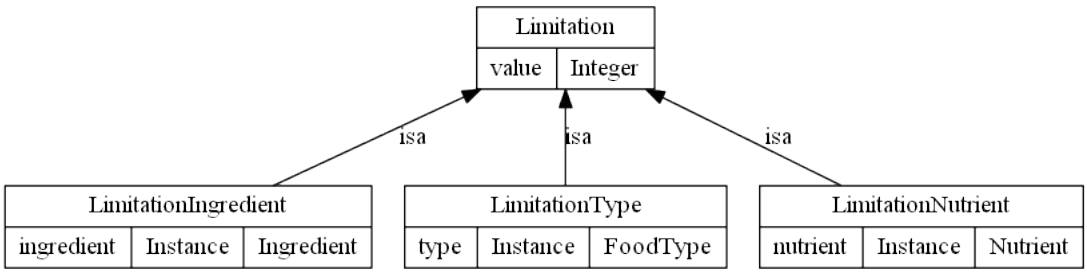

Figura 9: Clase *Limitation*

Para ser capaces de expresar una limitación o incluso una preferencia alimentícia, hemos hecho uso de esta clase para poder describir la voluntad de castigar o premiar el uso de ciertos alimentos o nutrientes.

Existen tres tipos (subclases) de la clase *Limitation* según el objetivo de la limitación:

- ***LimitationIngredient***: dirigido a influir en el uso de un ingrediente asociado
- ***LimitationType***: dirigido a influir en el uso de un tipo de ingrediente asociado (i.e. carne)
- ***LimitationNutrient***: dirigido a influir en el uso de los ingredientes que contengan el nutriente asociado

A parte de la instancia asociada según la subclase, las limitaciones también disponen del siguiente atributo:

- **value**: valor númerico asociado a la limitacion. Se usan valores negativos para castigar el uso de ingredientes y nutrientes, y valores positivos para premiarlo. La magnitud del valor indica la severidad con la que se quiere hacer efectiva la limitación. Un valor de 0 indica una limitación neutral y no tiene ningún efecto.

4.2. Método de resolución

En los problemas de SBC, donde se debe a partir de una preguntas obtener una solución adecuada, se realiza mediante una clasificación heurística. Los problemas que se resuelven mediante clasificación heurística se pueden dividir en tres partes:

4.2.1. Abstracción de los datos

En esta parte del problema pasaremos de un problema concreto a un problema abstracto. Obteniendo los datos de la persona (condiciones físicas, enfermedades, preferencias alimentarias...), el sistema abstraerá los datos necesarios por tal de poder ser usados luego para encontrar la mejor situación. En nuestro caso podemos encontrar esta parte del problema en los módulos NO ELS TINC A MÀ.

4.2.2. Asociación heurística

En esta parte pasaremos del problema abstracto a una solución abstracta. A partir de los diferentes datos del problema abstracto obtenido en la primera parte, obtendremos una solución abstracta, la cual consistirá en puntuar los diferentes platos según lo convenvenientes que sean para la persona. Por ejemplo una persona vegetariana obtendrá una puntuación menor para los platos que contengan carne. También en esta parte haremos el cálculo de Harris Benedict para obtener el número de calorías que necesita una persona al día. De esta forma tendremos una lista ordenada de los platos mas recomendados para cada usuario. Esta parte lo podemos encontrar en los módulos, FALTA MODULS.

4.2.3. Adaptación

Finalmente la Adaptación consiste en pasar de la solución abstracta a la solución concreta. Aquí cogeremos las listas de platos ordenadas i repartiremos estos platos en un menú dónde cada día se cumpla el número de calorías necesarias. Finalmente sacaremos por pantalla el menú obtenido para la persona i la cantidad de calorías que contiene. La adaptación esta formada por los módulos AAAAAAA.

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 ejecución 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

Con el intento de facilitar la resolución del problema y la lectura del código, este ha sido dividido de diferentes módulos que se van activando por conveniencia.

Dichos módulos son los siguientes:

- **Módulo MAIN**: es el módulo principal del sistema. Contiene una serie de funciones útiles para los otros módulos y la regla que inicia la ejecución del sistema.
- **Módulo ask_questions**: este módulo recopila toda la información que se necesita del usuario. Esta información se recopila mediante una serie de preguntas.
- **Módulo inference_of_data**:este módulo realiza todo el procesado de la información aportada por el ususario para poder generar la solución. En este módulo se calculan las calorías que

necesita el usuario y se les asigna la puntuación a los platos dependiendo de sus preferencias, enfermedades.

- **Módulo generate_solutions**: este módulo contiene las reglas que generan nuestro menú semanal.
- **Módulo printing**: la función de este módulo es printar la información al usuario y finalizar la ejecucción

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.