Práctica de Sistemas Basados en el Conocimiento
Inteligencia Artificial

Miquel Gómez Esteve
Guillem Rosselló Baiges
Jordi Foix Esteve
Mayo 2018

ÍNDICE

Identificación

Preferencias del usuario

El problema que tratamos en este caso es de recomendación de viajes a un usuario, por medio de unos parámetros preguntados previamente al mismo usuario. El enunciado sugería el uso de ciertos parámetros, que hemos intentado encuadrar en las preguntas que realizamos al usuario al inicio de la ejecución del código. La información que conseguimos es la siguiente:

- Longitud del viaje, número de días por ciudad y presupuesto aproximado (discretizado en tres categorías: bajo, regular y alto). Esta información es básica a la hora de estructurar el viaje, y en el caso concreto del presupuesto nos ayuda a descartar ciudades que están por encima del presupuesto escogido.

- El usuario va a querer realizar el viaje con un único medio de transporte o le es indiferente. En este caso hemos considerado modificar las sugerencias de preguntas del enunciado. De esta forma el transporte no es limitante para el usuario, en vez de esto ofrecemos al usuario la posibilidad de realizar un viaje temático ya se trate de un road trip, de un interrail o de un crucero (en el último caso hemos asumido que existe la posibilidad de realizar dichos trayecto a modo de crucero).

- Se trata o no de un viaje familiar. En caso de respuesta negativa, preguntamos si el viaje tiene que estar enfocado a un público concreto (de una edad determinada). Hemos optado por no escoger el viaje dependiendo de si se trata de una persona o de un grupo, pues consideramos que no es un parámetro que determine los sitios a visitar o los hoteles a los que hospedarse (en esta implementación del problema). En vez de esto tenemos los puntos de interés organizados según la edad de su público, y según si se trata o no de un punto de interés adecuado a familias.

- Se trata o no de un viaje de desconexión. En este caso hemos englobado varios aspectos del enunciado en estas preguntas . Así, las ciudades estan diferenciadas entre turísticas y alternativas según si el usuario busca, respectivamente, un viaje de

desconexión o un viaje turístico. Lógicamente, si el usuario selecciona ambas opciones en este caso concreto no descartaremos ninguno de los dos bloques de ciudades.

- Se trata o no de un viaje cultural. Este parámetro nos sirve de cara a recomendar puntos de interés al usuario, los puntos de interés cuentan con un slot que dice si este punto de interés es una atracción cultural o no.

Análisis de la viabilidad de construir el SBC

Este problema es un caso claro de problema a resolver mediante un sistema basado en el conocimiento. Hay dos elementos principales que lo justifican: por un lado el hecho que podemos organizar las posibles destinaciones como una base de conocimiento, generado por un usuario experto que lo incorpora al sistema. Por otro, lo que buscamos es restringir nuestra base de conocimiento en base a unas restricciones y preferencias introducidas por el usuario, de modo que la implementación mediante reglas se convierte en una de las mejores opciones de entre las posibles. Así, buscamos recorrer nuestra base de conocimiento seleccionando aquellos objetos que concuerden con las preferencias definidas por el usuario, problema que como hemos visto se adapta muy bien con la tipología de problemas que se solucionan mediante SBCs.

Descripción de los objetivos del problema y los resultados del sistema

El objetivo del proyecto es realizar la mejor recomendación de viaje posible a un usuario, usando como guía cierta información introducida al sistema por el usuario. Nuestra implementación da como solución una organización del viaje basada en los distintos días que durará. Asumiremos que el tiempo mínimo de visita de una ciudad es un día, de modo que cada ciudad tiene asignado un hotel en el que el usuario se hospedará. De esta manera cada día almacena información de la ciudad visitada, el hotel y los puntos de interés que se visitarán este día (obviamente pertenecientes a la ciudad visitada este día). Para que esta suposición se cumpla el sistema pide primero el número de días a visitar, y a continuación el número de ciudades limitado entre 1 y el número de días. El sistema ofrecerá dos posibilidades distintas de viaje al usuario, con ciudades totalmente diferentes.

Conceptualización

Conceptos del dominio

A continuación se describen brevemente los objetos de nuestro dominio. En el apartado Formalización se define la ontología y se entra en más detalle en los atributos de cada tipo de objeto y en las relaciones entre objetos.

City: Es el objeto base de nuestra ontología, define una ciudad particular e incluye los puntos de interés y los hoteles que se encuentran en esta ciudad. Guarda también información de las ciudades a las que se puede llegar des de la actual, organizadas según el/los medio/s de transporte a usar.

Interest Point:

Representa una atracción turística particular, y sus características respecto al público que recibe. Son los que definen la puntuación de cada ciudad, según si son o no de interés del usuario. De este modo tenemos una idea aproximada de cuales son las ciudades en las que el usuario encontraría más puntos de interés de su agrado.

Hotel: Representa propiamente un hotel y está clasificado según el precio de su estancia, discretizado también en tres categorías: lowCost, regular y expensive.

Subproblemas de la resolución

Con el objetivo de estructurar el proceso de resolución, hemos identificado distintas partes dentro de este, que nos harán más senzilla tanto la implementación como la lectura del código. A continuación se describen los distintos subproblemas:

· El primer paso de la resolución es seleccionar todas las ciudades y generar, para cada una, una instancia recomendación, con toda la información de la ciudad y una puntuación

por defecto de 0. Además, descartamos ya uno de los dos grandes bloques de ciudades en caso que sea necesario (ciudades alternativas o ciudades turísticas).

- A continuación calculamos dinámicamente la puntuación de cada ciudad, basándonos en la información entrada al sistema por el usuario y los atributos de cada ciudad (más adelante entraremos en detalle en el heurístico que usamos para escoger las ciudades recomendadas).

- Una vez las ciudades han sido valoradas asignamos ya la primera ciudad del viaje, de modo que la ciudad escogida es la que tiene una mejor puntuación de entre todas las opciones. Asignamos la primera ciudad del viaje.

- Repetimos el paso anterior, pero ahora usamos como heurístico la relación puntuación/ distancia a la ciudad anterior. Asignamos del mismo modo esta ciudad al viaje y repetimos el proceso hasta haber cumplido el número de ciudades escogido por el usuario.

- Una vez tenemos estas ciudades debemos seleccionar los puntos de interés que vamos a visitar. Para hacerlo, para cada ciudad ordenamos sus puntos de interés según la similitud con las preferencias del usuario. Una vez hecho, vamos asignando bloques de cuatro puntos de interés (número máximo de puntos de interés que se pueden visitar en un día) al viaje, empezando por la primera ciudad hasta la última y volver a empezar hasta llenar el tiempo total del viaje.

- El último paso está enfocado a generar un segundo viaje alternativo. Para ello buscamos que ninguna de las ciudades ya visitadas en el viaje anterior aparezcan también en este. Usaremos el filtrado de ciudades ya hecho en los pasos 1 y 2, a excepción de las ciudades anteriormente mencionadas, que serán eliminadas del conjunto. Una vez hecho este filtraje pasaremos otra vez a organizar los días del viaje.

Organización de los subproblemas

INFORMACIÓN

Este módulo se encarga de recolectar la información de las preferencias del usuario. Las reglas incluídas en este módulo son preguntas que reciben una respuesta y rellenan la instancia de la clase auxiliar Preferencia con la información del usuario.

PROCESSAT

En este módulo se calcula la puntuación de las ciudades, se genera una recomendación para cada ciudad y se eliminan las instancias de recomendación de ciudad que no concuerdan con la información introducida por el usuario.

CAMÍ

Este módulo recibe la información del módulo anterior, selecciona la primera ciudad a visitar y a partir de esta genera el viaje entero seleccionando ciudades adyacentes a la anterior.

PRESENTACIÓN

Finalmente el módulo de presentación recibe las dos opciones de viaje personalizado generadas con el módulo anterior y las muestra al usuario de forma clara por pantalla.

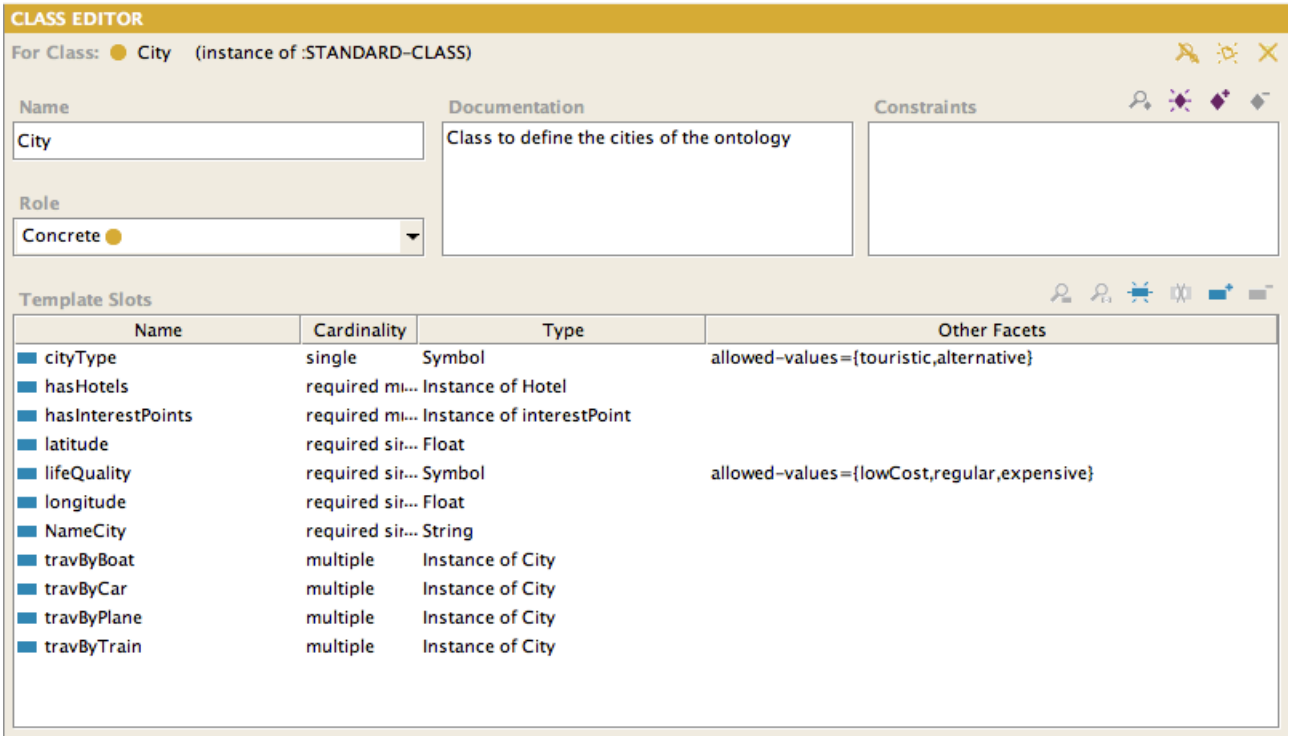
Formalización

Ontología

La ontología en este proyecto almacena toda la información en relación a las ciudades, hoteles y puntos de interés que hemos definido como dominio del problema.

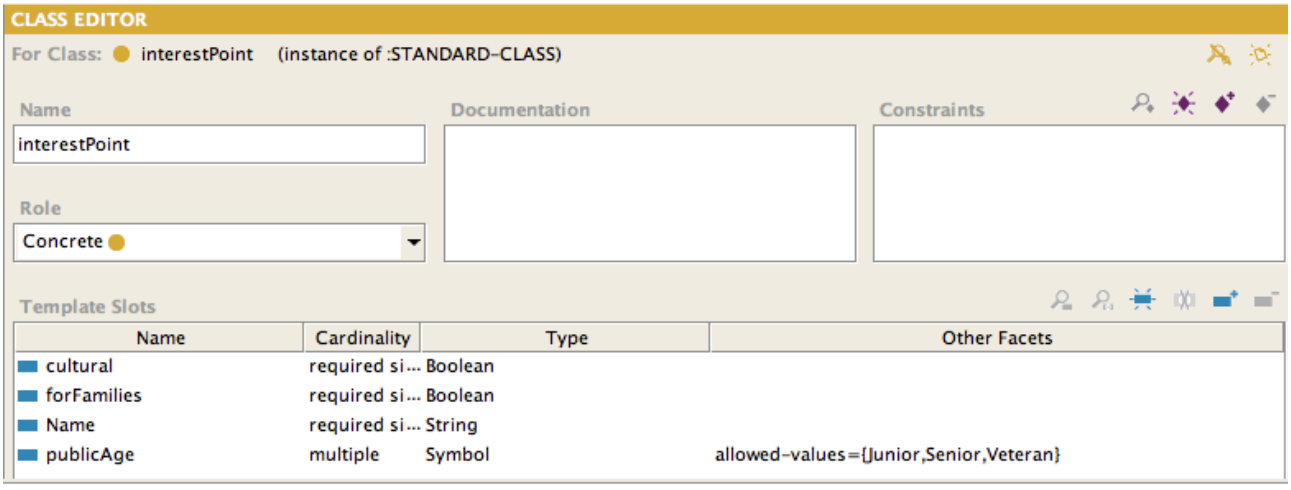
La clase principal de la ontología es City pues buscamos recomendar un grupo de ciudades al usuario, e incluye las instancias de los Hoteles y Puntos de Interés que se encuentran en dicha

ciudad. Cuenta con dos Slots definidos para diferenciar las ciudades y su calidad de vida entre sí: el parámetro *lifeQuality* nos da una aproximación discretizada del nivel de vida de la ciudad, diferenciando entre lowCost, regular y expensive, mientras *cityType* nos diferencia las ciudades turísticas de las alternativas. Cada instancia de City cuenta también con cuatro Slots que definen las ciudades a las que se puede llegar desde la actual, que son *byTrain*, *byPlane*, *byCar* y *byBoat*.



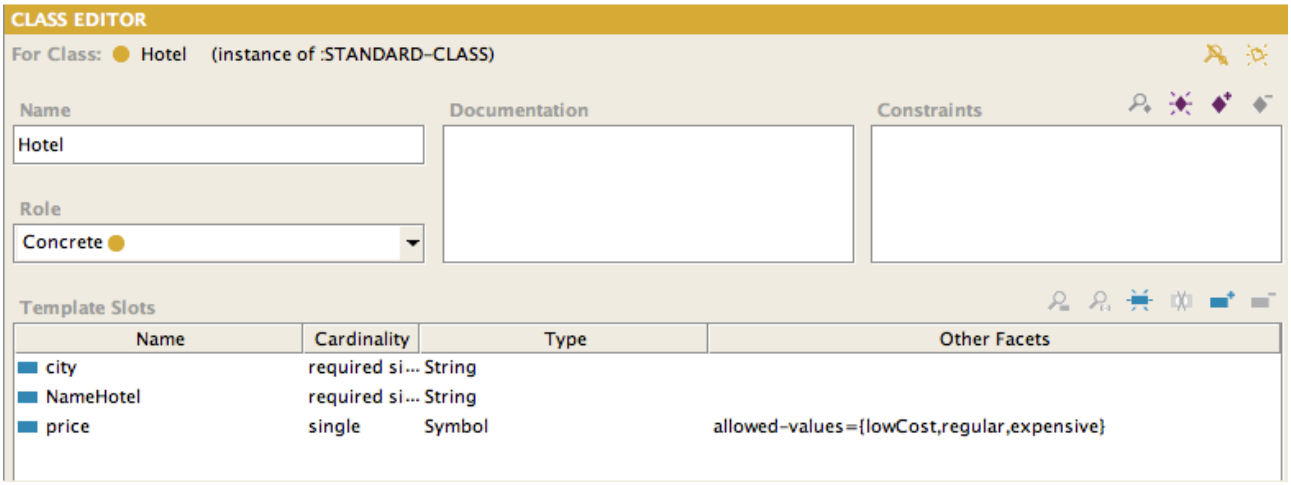
Definición de la clase City

La idea es pues que existe una relación para cada trayecto posible entre ciudades, encarada a facilitar la generación del viaje. De este modo recorreremos la ontología hasta llegar a la primera ciudad escogida, y a partir de esta seguiremos recorriendo la ontología mediante las relaciones descritas anteriormente (método de transporte), pasando de nodo ciudad a nodo ciudad.



Definición de la clase interestPoint

Los puntos de interés son los objetos que usaremos para definir la puntuación de las distintas ciudades. Para ello cuentan con un seguido de atributos: *cultural* y *forFamilies*, que respectivamente dicen si el punto de interés actual es de interés cultural y está adaptado para familias. A parte, tenemos tambien un multislot que nos da las franjas de edades para las que el punto de interés puede ser, valga la redundancia, de interés. Este multislot *publicAge* puede tener los valores Junior, Senior y Veteran.



Definición de la clase Hotel

Finalmente tenemos hoteles definidos por un nombre, una ciudad asociada y una aproximación del precio (una vez más discretizado a lowCost, regular y expensive). Los hoteles se asocian al viaje una vez la ciudad en la que se encuentra se escoge para el viaje, el usuario pasará toda su estancia en una ciudad en el mismo hotel y se buscará un hotel adaptado al presupuesto introducido por el usuario.

En vez de generar manualmente las instancias de ciudades, puntos de interés y hoteles del sistema, decidimos trabajar sobre un dominio inventado de forma arbitraria. Usamos un script para generar puntos de interés y hoteles aleatorios y asignarlos de forma aleatoria a las diferentes ciudades, de forma equilibrada y basandonos en las características de las ciudades. Para más detalle anexamos el script de generación y el fichero out.pins que contiene la mayor parte de las instancias usadas en el problema. Cabe decir que el script no da toda la información y tuvimos que introducir parte a mano, como por ejemplo la latitud y longitud de cada ciudad.

El proyecto en protegé se encuentra en el anexo, tanto los ficheros .pprj, .pins y .pont como un fichero .owl para visualizar la ontología usando un visualizador online.

Implementación

Clases extra

Hemos añadido dos clases auxiliares más al proyecto a parte de las clases ya definidas en Protegé: City_recomendation y ip_recomendation. La primera la usamos para hacer más sencilla la implementación, City_recomendation almacena la instancia de una ciudad recomendada y la puntuación asociada a esta ciudad según los parámetros de entrada del usuario. La utilizamos para ordenar las ciudades según orden de preferencia del usuario y poder escoger así aquellas ciudades que se acerquen más a los gustos del usuario.

La seguna, una vez seleccionadas las ciudades que se visitarán en el viaje, nos sirve para ordenar los puntos de interés de la ciudad por orden de similitud a las preferencias del usuario.

Funciones

En este proyecto hemos usado las funciones para facilitar el trabajo que se realiza en el módulo INFORMACIÓN. De este modo, cada función representa la estructura de una posible pregunta al usuario, y recibe como parámetro la información concreta que se pregunta al usuario.

Templates

Hemos creado dos templates: usuario y preferencias. La primera almacena la información relacionada directamente con el usuario, en nuestro caso el nombre y su ciudad natal (de hecho, buscamos la ciudad natal o la más cercana introducida en el sistema, ya que necesitamos partir y regresar a una ciudad origen conocida por el sistema). La seguna almacena las respuestas del usuario a las preguntas del sistema, que son información relacionada con las ciudades, los puntos de interés y los medios de transporte del sistema. Ambas templates son accedidas contínuamente durante la ejecución del código para seleccionar las ciudades más cercanas a los gustos del usuario.

Puntuación

Como hemos dicho con anterioridad, a la hora de seleccionar las ciudades usamos dos métodos heurísticos diferenciados según si se trata de la primera ciudad del viaje o del resto. El primer paso es valorar todas las ciudades según la similitud entre sus atributos, sus puntos de interés y las preferencias introducidas por el usuario. A continuación aparecen todas las reglas en las que se realiza un aumento de dicha puntuación. La ponderacione de las distintas reglas ha sido escogida de forma empírica después de realizar varios tests.

Así pues, para la primera ciudad del viaje seleccionamos la ciudad con mayor punutación despues de la valoración de las ciudades. Una vez tenemos la primera ciudad seleccionada, escogemos de entre aquellas ciudades a las que se puede llegar des de la ciudad actual (con los medios de transporte escogidos por el usuario), la ciudad con mejor ratio distancia/punutación. La distancia se calcula por medio de los parámetros longitud y latitud de las ciudades (distancia euclidiana), y lo hacemos de esta forma para facilitar el usuario el visitar ciudades cercanas en un mismo viaje y así no perder tiempo en transporte. De este modo al usuario se le ofrecerá un viaje a la ciudad que se acerca más a sus preferencias, y a las ciudades cercanas con las que, una vez más, las preferencias concuerden. Estas son las reglas usadas para calcular la distancia y valorar el ratio distancia/puntuación.

Una vez escogidas las ciudades que compondrán el viaje, hacemos una valoración heurística de los puntos de interés de dichas ciudades. Estos son ordenados según su similitud con las preferencias del usuario, y asignados a los días de visita considerando que en un día se pueden visitar como mucho cuatro puntos de interés. Aquí estan las reglas usadas para valorar los puntos de interés.

Proceso de implementación y prototipos previos

El primer diseño de la ontología que realizamos contaba con una clase más, la clase continente. Pensamos en un primer momento que nos sería útil contar con esta subdivisión, pero vimos rápidamente que los diferentes métodos de transporte y el uso del slot distancia dejaban

obsoleta esta opción. Eliminamos esta clase y nos quedamos definitivamente con City, interestPoint y Hotel.

Al plantear la valoración de las ciudades, pensamos usar en un primer momento la clase auxiliar interestPoint_recomendation. Así, almacenábamos los puntos de interés que se adecuaban a las preferencias del usuario para posteriormente organizar el viaje. Vimos a la hora de implementar el sistema que era mucho mejor valorar directamente las ciudades, almacenarlas con la clase City_recomendation y una vez asignadas las ciudades al viaje, seleccionar los puntos de interés.

Tuvimos problemas con las distancias, definíamos *latitude* y *longitude* como floats pero CLIPS nos daba problemas en los valores negativos. Encontramos que el rango de valores en los que se encuentran latitud y longitud son [-90, 90] y [-180, 180] respectivamente. Para evitar trabajar con valores negativos y dado que trabajamos con distancias, sumamos 90 a *latitude* y 180 a *longitude* (harcoded).

JUEGOS DE PRUEBA

Preguntas del sistema al usuario:

Quin és el nom de l'usuari?
Quina és la ciutat de l'usuari?

Quants dies ha de durar el viatge? (1, 30)
Quantes ciutats es volen visitar?" (1, #dies)
Quin és el pressupost del viatge? (baix, mitja, alt)

El viatge es realitzarà únicament en cotxe?
El viatge es realitzarà únicament en tren?
El viatge es realitzarà únicament en vaixell?
(de estas tres preguntas solo una puede tener una respuesta afirmativa)

Es tracta d'un viatge familiar?
A quin public esta orientat el viatge?
(la segunda pregunta solo se considera en caso que la respuesta a la primera sea negativa)

Es tracta d'un viatge de desconnexio?
Es tracta d'un viatge cultural?

Juego de prueba 1

Quin és el nom de l'usuari? **Guillem**
Quina és la ciutat de l'usuari? **"Barcelona"**

Quants dies ha de durar el viatge? (1, 30) ***4***
Quantes ciutats es volen visitar?" (1, #dies) **3**
Quin és el pressupost del viatge? (baix, mitja, alt) **baix**

El viatge es realitzarà únicament en cotxe? **no**
El viatge es realitzarà únicament en tren? **no**
El viatge es realitzarà únicament en vaixell? **no**

Es tracta d'un viatge familiar? **no**
A quin public esta orientat el viatge? **Junior**

Es tracta d'un viatge de desconnexio? **si**
Es tracta d'un viatge cultural? **si**

Juego de prueba 2

Quin és el nom de l'usuari? **Guillem**

Quina és la ciutat de l'usuari? **"Madrid"**

Quants dies ha de durar el viatge? (1, 30) ***25***
Quantes ciutats es volen visitar?" (1, #dies) **1 5**
Quin és el pressupost del viatge? (baix, mitja, alt) **alt**

El viatge es realitzarà únicament en cotxe? **no**
El viatge es realitzarà únicament en tren? **no**
El viatge es realitzarà únicament en vaixell? **no**

Es tracta d'un viatge familiar? **si**

Es tracta d'un viatge de desconnexio? **no**
Es tracta d'un viatge cultural? **si**

Juego de prueba 3

Quin és el nom de l'usuari? **Guillem**

Quina és la ciutat de l'usuari? **"Venice"**

Quants dies ha de durar el viatge? (1, 30) ***2***
Quantes ciutats es volen visitar?" (1, #dies) **1**
Quin és el pressupost del viatge? (baix, mitja, alt) **mitja**

El viatge es realitzarà únicament en cotxe? **no**
El viatge es realitzarà únicament en tren? **no**
El viatge es realitzarà únicament en vaixell? **si**

Es tracta d'un viatge familiar? **no**
A quin public esta orientat el viatge? **Veteran**

Es tracta d'un viatge de desconnexio? **si**

Es tracta d'un viatge cultural? no

Juego de prueba 4

Quin és el nom de l'usuari? **Guillem**

Quina és la ciutat de l'usuari? **"Sidney"**

Quants dies ha de durar el viatge? (1, 30) ***10***
Quantes ciutats es volen visitar?" (1, #dies) **20**
Quin és el pressupost del viatge? (baix, mitja, alt) **baix**

El viatge es realitzarà únicament en cotxe? **si**

Es tracta d'un viatge familiar? **no**
A quin public esta orientat el viatge? **Junior**

Es tracta d'un viatge de desconnexio? **no**
Es tracta d'un viatge cultural? **si**

Juego de prueba 5

Quin és el nom de l'usuari? **Guillem**

Quina és la ciutat de l'usuari? **"Barcelona"**

Quants dies ha de durar el viatge? (1, 30) ***10***
Quantes ciutats es volen visitar?" (1, #dies) **5**
Quin és el pressupost del viatge? (baix, mitja, alt) **alt**

El viatge es realitzarà únicament en cotxe? **no**
El viatge es realitzarà únicament en tren? **si**

Es tracta d'un viatge familiar? **no**
A quin public esta orientat el viatge? **Veteran**

Es tracta d'un viatge de desconnexio? **si**
Es tracta d'un viatge cultural? **si**

Juego de prueba 6

Quin és el nom de l'usuari? **Guillem**

Quina és la ciutat de l'usuari? **"Barcelona"**

Quants dies ha de durar el viatge? (1, 30) ***1***
Quantes ciutats es volen visitar?" (1, #dies) **1**
Quin és el pressupost del viatge? (baix, mitja, alt) **alt**

El viatge es realitzarà únicament en cotxe? **no**
El viatge es realitzarà únicament en tren? **si**

Es tracta d'un viatge familiar? **si**
A quin public esta orientat el viatge? **Adult**

Es tracta d'un viatge de desconnexio? **no**
Es tracta d'un viatge cultural? **si**