Recomendador de Actividades Turísticas

MASTER EN TECNOLOGÍAS DE LA INFORMACIÓN Y SISTEMAS INFORMÁTICOS

Alumno: Luis Martínez Marina
Tutora: Estefanía Martín Barroso
UNIVERSIDAD REY JUAN CARLOS MADRID

RESUMEN

El acceso a la información sobre las actividades realizadas en una determinada ciudad resulta a veces complejo debido a la gran diversidad ofertada en la actualidad, por lo que resulta necesario dotar al usuario de algunas ayudas para que pueda acceder a dicha información de un modo más sencillo y ordenado.

Para la selección de las actividades se tienen en cuenta diversos factores, como son la disponibilidad horaria, el precio, la accesibilidad a discapacidades, la zona en la que se encuentra y los medios de transporte públicos para llegar a dicha actividad.

La realización de una ontología sobre las actividades, sus medios de transporte y la accesibilidad proporciona al proyecto una mayor simplificación para luego realizar el sistema experto que realiza las asignaciones de las actividades. Para ello se utiliza el software *Protégé* y el sistema experto CLIPS.

Para la asignación de las actividades, el usuario elige entre los diferentes grados de afición a los tipos de actividades seleccionadas, su edad, presupuesto y disponibilidad horaria.

La implementación de una aplicación Web hace que el usuario pueda utilizar la aplicación desde cualquier lugar a través de la tecnología .NET.

Además, se crea un sistema de recomendaciones basado en el filtrado colaborativo, para añadir más ofertas al usuario basado en la experiencia pasada de otros usuarios.

INDICE

1. INTRODUCIÓN 1

2. ESTADO DEL ARTE 3

2.1 ASP.NET 3

2.1.1 Elementos de la estructura .NET 4

2.1.2 Entorno común de ejecución 4

2.1.2.1 Sistema de tipos común 5

2.1.3 Biblioteca de clases .NET 6

2.1.4 Componentes de unificación 7

2.1.5 ASP.NET 7

2.1.5.1 Formularios Web 7

2.1.5.2 Servicios Web 8

2.1.6 Formularios de Windows 8

2.1.7 Visual Studio .NET 8

2.2 SISTEMAS EXPERTOS (CLIPS) 11

2.2.1 Apuntes históricos 11

2.2.2 Características Principales 13

2.2.2.1 HECHOS EN CLIPS 13

2.2.2.2 REGLAS 13

2.2.3 ORIENTACIÓN A OBJETOS EN CLIPS 14

2.3 SISTEMAS DE RECOMENDACIÓN 17

2.3.1 CLASIFICACION GENERAL 17

2.3.2 PROBLEMAS GENERALES 19

2.3.3 SISTEMAS BASADOS EN EL CONTENIDO 21

2.3.3.1 PROBLEMAS GENERALES 24

2.3.3.2 TENDENCIAS GENERALES 24

2.3.4 SISTEMAS DE FILTRADO COLABORATIVO 24

2.3.4.1 HISTORIA 25

2.3.4.2 CLASIFICACIÓN 25

a) BASADA EN LA MEMORIA 26

b) BASADAS EN EL MODELO 27
2.3.4.3 PROBLEMAS 28
2.3.4.4 TENDENCIAS GENERALES 29
2.3.5 RECOMENDACIONES BASADAS EN EL CONOCIMIENTO 29
2.3.5.1 FUNCIONAMIENTO 30
2.3.5.2 CONCLUSIONES 31
2.3.6 CONCLUSIONES A LOS SISTEMAS DE RECOMENDACIÓN 31
3. ANÁLISIS DE REQUISITOS 33
3.1 Diseño Ontología 33
3.2 Diseño Base de Datos 38
3.3 Diseño Aplicación Web 40
3.3.1 ELEMENTOS 41
3.3.2 PRUEBAS DE DISEÑO 43
4. IMPLEMENTACIÓN 49
4.1 CLIPS 49
4.2 APLICACIÓN WEB 51
4.2.1 Funcionamiento Básico 53
4.2.2 RECOMENDACIONES 57
4.2.3 Generación Informe PDF 59
4.3 PRUEBAS 61
4.3.1 VALIDACIÓN FORMULARIO 61
4.3.2 PRUEBAS DE LA APLICACIÓN 63
4.4 VALIDACIÓN HMTL Y CSS 68
4.5 VISION EN OTROS NAVEGADORES 70
4.5.1 INTERNET EXPLORER 8 70
4.5.2 GOOGLE CHROME 5.0 72
4.5.3 MOZILLA FIREFOX (3.5.7) 73
4.5.4 OPERA (10.53) 74
4.5.5 SAFARI 5.0 75
5. CONCLUSIONES Y TRABAJOS FUTUROS 77
6. BIBLIOGRAFÍA 79
APÉNDICE A: MANUAL DE USUARIO 81

ÍNDICE DE ILUSTRACIONES

Ilustración 1: Componentes de la Estructura .NET 4
Ilustración 2: Organización de la Biblioteca de clases .NET 6
Ilustración 3: La estructura .NET 7
Ilustración 4: Diseño Ontología 35
Ilustración 5: Introducción Instancias 36
Ilustración 6: Diseño Base de Datos 39
Ilustración 7: Página 1 Prueba 1 43
Ilustración 8: Página 1 Prueba 2 44
Ilustración 9: Página 2 Prueba1 44
Ilustración 10: Página 2 Prueba 2 45
Ilustración 11: Página 2 Prueba 3 46
Ilustración 12: Página 2 Prueba 4 46
Ilustración 13: Pasos de la Aplicación 49
Ilustración 14: Funcionamiento Aplicación 54
Ilustración 15: Ejemplo Generación Documento PDF 60
Ilustración 16: Inserción Datos Personales 61
Ilustración 17: Elección Grados de Afición 62
Ilustración 18: Disponibilidad Horaria 63
Ilustración 19: Validación Código HTML 68
Ilustración 20: Validación Hoja de estilo 69
Ilustración 21: Internet Explorer 8 Página 1 70
Ilustración 22: Internet Explorer 8 Página 2 71
Ilustración 23: Google Chrome Página 1 72
Ilustración 24: Google Chrome Página 2 72
Ilustración 25: Firefox Página 1 73
Ilustración 26: Firefox Página 2 73
Ilustración 27: Opera Página 1 74
Ilustración 28: Opera Página 2 74
Ilustración 29: Safari Página 1 75
Ilustración 30: Safari Página 2 75

1. INTRODUCIÓN

En la actualidad existe una gran variedad de actividades turísticas para que una persona realice, quizás demasiadas, con información poco precisa, mal organizada con actividades que un usuario no desea realizar porque no es adecuada para él, por edad o por no estar adaptada para su discapacidad si este sufriese alguna.

Las numerosas técnicas de recomendación usadas por diferentes portales, son una gran ayuda y un punto de partida para este proyecto, ya que a través de diversos métodos de recomendación, muestra gran variedad actividades según el precio, cercanía, valoración de otros usuarios, relación calidad-precio o simplemente por publicidad que esta actividad haya pagado. La accesibilidad tanto a edificios como a espacios al aire libre por parte de personas con discapacidad o movilidad reducida es una de las prioridades que debemos tener en cuenta, ya que se debe facilitar el acceso a todos los edificios ya sean de interés cultural o no.

En este proyecto, se pretende crear una aplicación Web, en la que a partir de las preferencias del usuario, sus datos personales, su discapacidad (si padeciese alguna) y la disponibilidad horaria se le ofrezcan las propuestas culturales que se ajusten más a sus preferencias. Para ello se utilizarán diferentes técnicas y herramientas de índoles muy diversas.

La utilización de técnicas de Inteligencia Artificial a través del lenguaje de sistemas expertos CLIPS, supondrá un apoyo básico para el desarrollo del proyecto, ya que con dicho lenguaje se crearán las diferentes recomendaciones turísticas.

Para albergar dicha aplicación, se utilizará la tecnología ASP.NET 2.0, que proporciona una gran potencia de servicio, ya que permite crear aplicaciones web dinámicas, con una gran capacidad de cómputo, que da la posibilidad de poder programar el comportamiento de todos los elementos como si se tratase de una aplicación de escritorio, además de poder incluir información a través de una base de datos

Por último, la creación de un sistema de recomendación basado en la experiencia de otros usuarios hace mucho más completa a la aplicación, ya que el usuario puede tener recomendaciones personalizadas acorde con sus preferencias.

Este proyecto, se limitará simplemente a una ciudad, como es Móstoles, ya que es una zona con una población importante, para disponer de una variedad considerable de actividades, pero sin que sean excesivas las recomendaciones insuficientes dada la gran cantidad de ofertas de ocio que ofrece. El tamaño esta ciudad es adecuado para el desarrollo de este proyecto.

En definitiva, se basa en facilitar al usuario la posibilidad de conocer y planificar diferentes actividades turísticas dentro de una ciudad en concreto de acuerdo a sus preferencias y necesidades.

A continuación, se detalla la estructura general de esta memoria del trabajo de fin de Máster:

- En este capítulo se ha hecho una pequeña introducción al proyecto.
- El capítulo 2 es el estado del arte en lo que respecta a las distintas tecnologías y técnicas a utilizar en este proyecto (ASP.NET, CLIPS y Sistemas de Recomendación).
- El capítulo 3 muestra el análisis de requisitos.
- El capítulo 4 detalla la implementación del proyecto en diversos aspectos (el código CLIPS, la aplicación Web, las diferentes pruebas al sistema, la validación del código HMTL y CSS y la visión en diferentes navegadores).
- En el capítulo 5 se describen las conclusiones al proyecto y líneas futuras.
- Por último, se añade un apéndice con un manual de usuario para una mejor utilización de la aplicación.

2. ESTADO DEL ARTE

2.1 ASP.NET

La revolución de Internet tuvo lugar a finales de los años 90 y supuso un drástico cambio en la forma en la que los individuos y las empresas se comunicaban entre sí. La estructura .NET representa un conjunto de servicios y bibliotecas unificado y orientado a objetos que engloba el nuevo papel de los programas basados y dirigidos a la red. De hecho, esta estructura es la primera plataforma diseñada esencialmente con Internet como base.

La estructura .NET ofrece distintas ventajas que pueden ser utilizadas por los programadores:

- Un modelo de programación consistente.
- Compatibilidad directa con cuestiones de seguridad.
- Operaciones simplificadas de desarrollo.
- Sencilla implementación y mantenimiento de las aplicaciones.

Los distintos lenguajes de programación ofrecen diferentes modelos para realizar la misma operación. Una misma acción puede realizarse por diferentes lenguajes, como son Visual Basic y C#, por este motivo, los conocimientos de los programadores deben ser muy completos.

La biblioteca de clases de .NET es una de las piezas básicas de la estructura .NET, que en ocasiones se denomina *Biblioteca de Clases Base* (BCB). Está compuesta por cientos de clases que se pueden utilizar en los siguientes casos:

- Procesamiento de XML.
- Manipulación de datos de distintos orígenes.
- Depuración de código y trabajo con registros de eventos.
- Flujos de datos y archivos.
- Gestión del entorno de ejecución.
- Desarrollo de servicios Web, componentes y aplicaciones estándar de Windows.
- Parcelas de seguridad.

- Servicios de directorios.

La funcionalidad de esta biblioteca se puede encontrar en todos los lenguajes de .NET, con lo que se genera un consistente modelo de objetos independientemente del lenguaje de programación utilizado.

2.1.1 Elementos de la estructura .NET

La estructura .NET se compone de tres elementos fundamentales:

- Entorno común de ejecución
- Biblioteca de clases de .NET
- Componentes de unificación

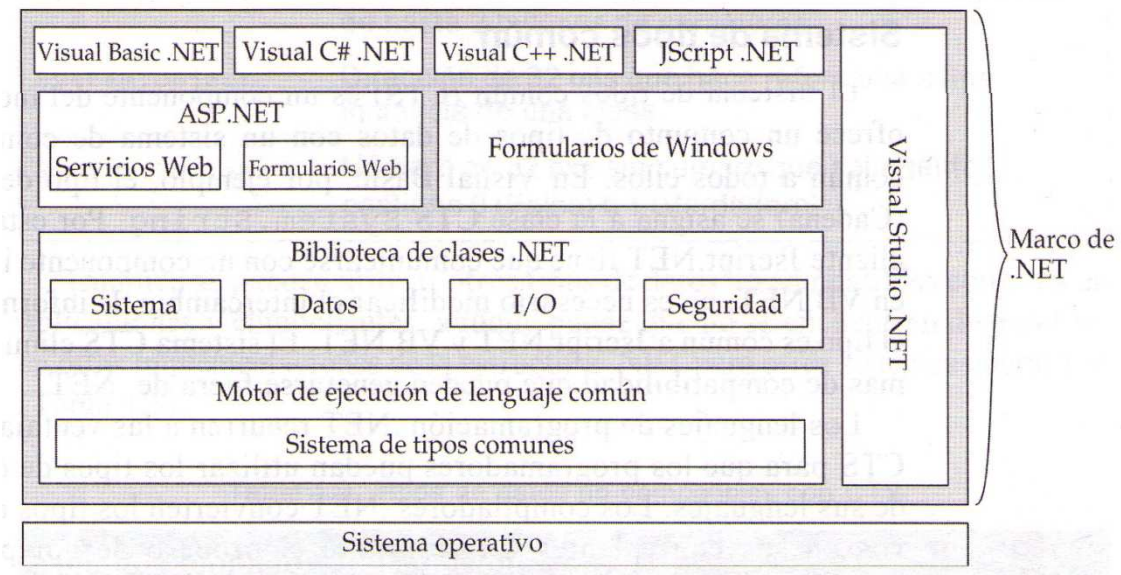


Ilustración 1: Componentes de la Estructura .NET

2.1.2 Entorno común de ejecución

El *Entorno Común de Ejecución* (CLR) es el nivel situado entre una aplicación y el sistema operativo en el que se ejecuta. Este entorno simplifica el diseño de la aplicación y reduce la cantidad de código ya que se ofrece una serie de servicios de ejecución que incluye la gestión de memoria, de hilos, de duración de los componentes y el procesamiento predeterminado de errores. La principal ventaja de este entorno es que proporciona estos servicios de ejecución a todas las aplicaciones,

independientemente del lenguaje de programación en el que se haya creado y sin que el programador tenga que realizar operaciones adicionales.

El entorno común de ejecución se encarga de compilar el código antes de ejecutarlo. En lugar de generar una representación binaria del código como los compiladores tradicionales, .NET genera una representación en un lenguaje común a la estructura .NET: el Lenguaje Intermedio de Microsoft (MSIL), también denominado IL. Cuando se ejecuta un código por primera vez, el CLR invoca un compilador especial denominado JIT, que transforma el lenguaje IL en instrucciones ejecutables específicas al tipo y modelo del procesador del sistema. Como todos los lenguajes .NET disponen de la misma representación compilada, sus características de rendimiento serán idénticas. Esto significa que un programa escrito en Visual Basic .NET puede funcionar del mimo modo que un programa diseñado en Visual C++ .NET por ejemplo.

2.1.2.1 Sistema de tipos común

El *Sistema de Tipos Común* (CTS) es un componente del motor CLR, que ofrece un conjunto de tipos de datos con un sistema de comportamientos común a todos ellos. En Visual Basic, por ejemplo, el tipo de datos String se asigna a la clase CTS System.String. Por esta razón, si un cliente Jscript .NET tiene que comunicarse con un componente implementado en VB .NET, no es necesario modificar el intercambio de información ya que el tipo es común a Jscript .NET y VB.NET. El sistema CTS elimina los problemas de compatibilidad que pueden generarse fuera de .NET.

Los lenguajes de programación .NET recurren a las ventajas del sistema CTS para que los programadores puedan utilizar los tipos de datos propios de sus lenguajes. Los compiladores .NET convierten los tipos de datos propios de sus lenguajes. Los compiladores .NET convierten los tipos de datos nativos en sus equivalentes CTS durante el proceso de compilación. Los programadores también pueden utilizar directamente los tipos CTS en el código que diseñen.

También se pueden utilizar otros tipos de datos no compatibles con CTS en aplicaciones y componentes, aunque puede que no se encuentren disponibles en otras implementaciones de la estructura .NET para otros sistemas operativos.

2.1.3 Biblioteca de clases .NET

Como ya se ha indicado anteriormente, la biblioteca de clases de .NET contiene cientos de clases que modelan el sistema y los servicios que ofrece. Para que resulte más sencillo de entender y de manejar, se divide en espacios de nombre. El espacio de nombre raíz de la biblioteca de clases .NET se denomina System y contiene clases principales y tipos de datos como Int32, Object, Array y Console. Los espacios de nombre secundarios se adjuntan al nombre System.

A continuación se muestra algunos ejemplos de espacio de nombres anidados.

- System.Diagnostic: incluye clases para trabajar con el Registro de eventos
- System.Data: facilita el manejo de datos de diferentes fuentes de datos (System.Data.OleDB se encuentra en este espacio de nombre y contiene las clases ADO.NET)
- System.IO: contiene clases para trabajar con flujos de archivos y datos.

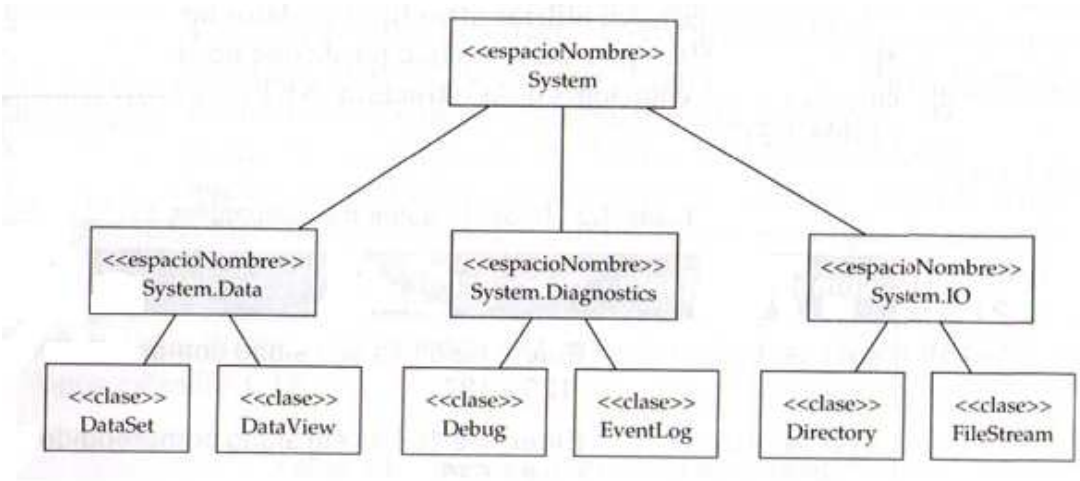


Ilustración 2: Organización de la Biblioteca de clases .NET

La figura 2 muestra la relación entre algunos principales espacios de nombres en la biblioteca de clases .NET.

Entre las ventajas de la biblioteca de clases .NET podemos encontrar un consistente conjunto de servicios disponibles para todos los lenguajes .NET y una implementación simplificada, ya que esta biblioteca se utiliza en todas las implementaciones de la estructura .NET

2.1.4 Componentes de unificación

Los componentes de unificación permiten acceder a los servicios que ofrece esta estructura:

- ASP.NET.
- Formularios de Windows.
- Visual Studio .NET.

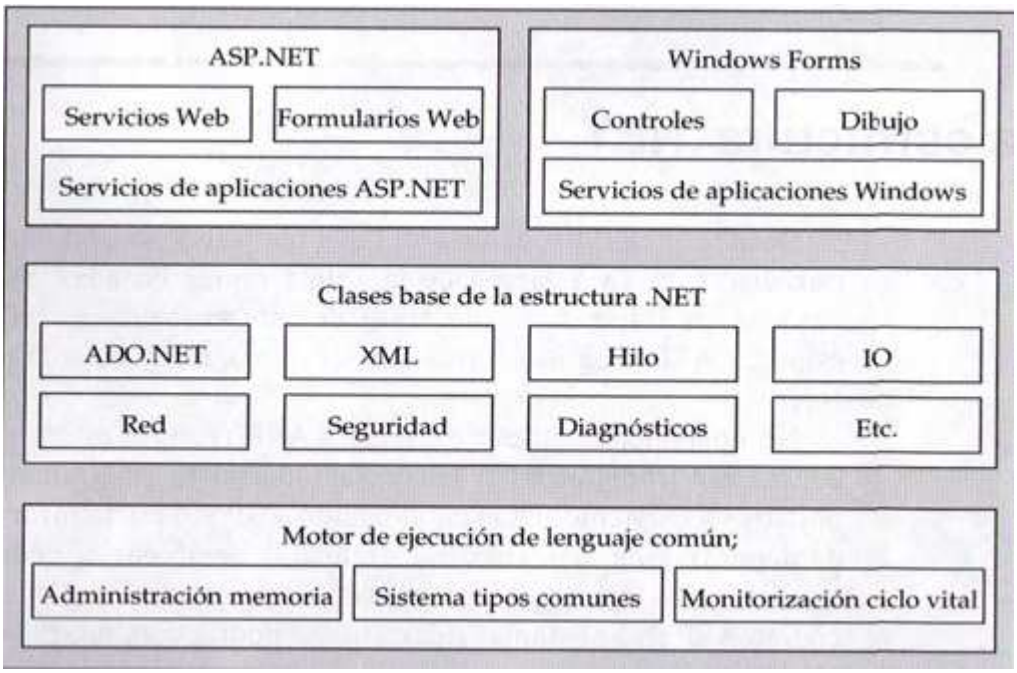


Ilustración 3: La estructura .NET

2.1.5 ASP.NET

El componente ASP.NET ofrece dos elementos principales: formularios Web y servicios Web.

2.1.5.1 Formularios Web

Los formularios Web simplifican el desarrollo Web para que incluso se pueda arrastrar y soltar los controles sobre el diseño que se utilice para editar una página y,

de esta forma crear aplicaciones Web interactivas. La información se puede validar del lado del cliente o del lado del servidor.

2.1.5.2 Servicios Web

Un servicio Web es una aplicación que muestra una interfaz a través de métodos de acceso estándar. Este tipo de servicios se pueden utilizar por otras aplicaciones y componentes y su diseño no está dirigido al uso directo por usuarios humanos. Facilitan el diseño de aplicaciones que integran elementos de recursos remotos. Por ejemplo, se puede escribir un Servicio Web que ofrezca información meteorológica a los suscriptores del servicio en lugar de que los suscriptores tengan que enlazar a una página o descargar un archivo de un sitio Web. Basta con que los clientes invoquen un método del servicio Web del mismo modo que si se tratara de un método de un componente instalado de un sistema. De esta forma, pueden acceder a la información meteorológica disponible en un formato de sencillo manejo que pueden integrar en sus propias aplicaciones o páginas Web personales.

2.1.6 Formularios de Windows

El nombre formularios de Windows hace referencia al conjunto unificado de clases compatibles con la creación de aplicaciones personales, aplicaciones que disponen de una interfaz gráfica de usuario (GUI). Este tipo de componentes facilita el diseño de aplicaciones destinadas al usuario que utilizan cualquier lenguaje de programación .NET. Es más, con ayuda de Visual Studio .NET, los programadores pueden diseñar de forma sencilla formularios por medio de las técnicas de arrastrar y soltar.

2.1.7 Visual Studio .NET

Se trata de un entorno de desarrollo apto para todos los lenguajes. Visual Studio .Net simplifica las labores de desarrollo en un entorno multilenguaje a través de funciones de depuración extremo a extremo entre todos los lenguajes de programación; diseños visuales para XML, HTML, datos y código, servidor y compatibilidad IntelliSense (cumplimiento de instrucciones). Visual Studio .NET sustituye a los entornos de desarrollo Visual Basic 6, Visual C++ y Visual InterDev.

Este entorno consigue tal nivel de integración ya que se basa en las funciones de la estructura .NET. El diseño de formularios Web y formularios de Windows mejora la productividad del proceso de desarrollo. La integración de las funciones de implementación mejora la productividad durante el proceso de depuración posterior a la implementación.

2.2 SISTEMAS EXPERTOS (CLIPS)

A modo de introducción se puede decir que CLIPS es una herramienta que provee un ambiente de desarrollo para la producción y ejecución de sistemas expertos. CLIPS es un acrónimo de *C Language Integrated Production System* (Sistema de Producción Integrado en Lenguaje C). En la actualidad, entre los paradigmas de programación que soporta CLIPS se encuentran la Programación lógica, la Programación imperativa y la Programación Orientada a Objetos. CLIPS probablemente es el sistema experto más ampliamente usado debido a que es rápido, eficiente y gratuito. Aunque ahora es de dominio público, aún es actualizado y mantenido por su autor original, Gary Riley.

2.2.1 Apuntes históricos

Durante el año 1984, en el Lyndon B. Johnson Space Center, la sección de Inteligencia Artificial había desarrollado alrededor de una docena de prototipos de sistemas expertos usando *hardware* y *software* de aquella época. A pesar del demostrado potencial de los sistemas expertos, la mayoría de aquellos prototipos no estaban siendo usados regularmente. Según la NASA esto se debió a que el lenguaje de programación usado para el desarrollo de estas aplicaciones era LISP. Se encontraron varias debilidades de LISP, entre las que se destacan tres: no estaba disponible para una amplia variedad de equipos de cómputo, no era fácilmente integrable con otras aplicaciones, y su costo era muy elevado.

La sección de Inteligencia Artificial notó que el uso de un lenguaje convencional (como C) eliminaría la mayoría de esos inconvenientes. Al principio se buscó que la nueva herramienta fuera desarrollada por terceros, pero los costos aún eran demasiado altos y los tiempos de entrega no eran satisfactorios. Por lo tanto, fue necesario que la sección de Inteligencia Artificial desarrollara su propio lenguaje para sistemas expertos basado en C.

El primer prototipo de CLIPS fue desarrollado durante 1985, y tomó poco más de dos meses. Era compatible con todos los desarrollos hechos por la sección de Inteligencia Artificial, y su sintaxis estaba basada en la sintaxis de ART (otra herramienta para el desarrollo de sistemas expertos). Sin embargo, CLIPS fue desarrollado sin tener acceso o haber conocido previamente el código fuente de ART.

Inicialmente, con el desarrollo de CLIPS se buscaba tener mayor conocimiento sobre la construcción de sistemas expertos y sentar las bases de un lenguaje para reemplazar las herramientas comerciales que estaban siendo usadas. La versión 1.0 de CLIPS demostró que eso era posible. Después de un desarrollo adicional, se vio que el costo de CLIPS sería significativamente menor al de otras herramientas y que sería ideal para entrenamiento. Otro año de desarrollo y de uso interno sirvió para mejorar portabilidad, desempeño, funcionalidad y documentación de soporte. A mediados de 1986, CLIPS v3.0 estuvo disponible para grupos fuera de la NASA.

Originalmente, CLIPS era un lenguaje de reglas basado en el Algoritmo Rete (Programación Lógica). CLIPS v5.0 (1991) introdujo dos nuevos paradigmas de programación: Programación Imperativa y Programación Orientada a Objetos (POO). El lenguaje POO dentro de CLIPS es llamado COOL (CLIPS Object-Oriented Language, o Lenguaje Orientado a Objetos de CLIPS). CLIPS v5.1 (1991) ya soportaba las recientemente desarrolladas o mejoradas interfaces X Window, MS-DOS y Macintosh. CLIPS v6.0 (1993) tenía nuevas funcionalidades relacionadas con el reconocimiento de patrones en objetos/reglas y soporte a Ingeniería de Software basada en reglas. CLIPS v6.1 (1998) soportaba compiladores C++, aunque ya no soportaba los viejos compiladores C no ANSI. También se agregaron comandos para llevar control del tiempo de desarrollo y para funciones definidas por el usuario. CLIPS v6.2 es soportada por diversos sistemas operativos, y tiene mejoras en su interfaz de desarrollo para Windows 95/98/NT y MacOS. Actualmente, CLIPS es mantenido fuera de la NASA como software de dominio público y va por su versión 6.30.

Existen derivados de CLIPS para ser usados en otros lenguajes como por ejemplo:

- JESS: implementación de CLIPS en Java+.
- FuzzyCLIPS: incorpora a CLIPS la posibilidad de usar razonamiento difuso.
- CLIPSMM: una interfaz libre de CLIPS con C++.
- PHLIPS: extensión para PHP.
- PyCLIPS: interfaz libre para Python.
- EHSIS: Implementación del lenguaje CLIPS con APIs adicionales y documentación en castellano.
- CLIPSNET: librería de C#, (utilizada en este proyecto).

2.2.2 Características Principales

Como ya se ha indicado, CLIPS es un lenguaje que permite la representación de conocimiento declarativo y procedimental. Además:

- Permite representar reglas de producción y *frames*.
- La base es un motor de inferencias con razonamiento hacia delante.
- El motor de inferencias esta implementado sobre un intérprete del lenguaje.
- Los dos elementos que permiten representar problemas utilizando reglas son los hechos y las reglas.

2.2.2.1 HECHOS EN CLIPS

Los hechos en CLIPS pueden ser de dos tipos: *ordered facts* y *deftemplate facts*.

Un ejemplo de *ordered facts* es:

```
(padre juan pedro)
(num-hijos juan 2)
```

Un ejemplo de *deftemplate* es:

```
(deftemplate persona
     (slot nombre(type STRING))
     (slot edad(type INTEGER)(default 0)))
```

La creación de hechos se realiza mediante las sentencias *assert* (uno solo) o *deffact* (un conjunto), por ejemplo:

```
(assert(padre pepe juan))
(assert(persona(nombre"pedro")(edad 25)))
(deffacts mis-hechos
     (casa roja)(pelota verde)
     (persona(nombre “luis”)(edad 33)))
```

2.2.2.2 REGLAS

Las reglas en CLIPS están formadas por:

- Una parte izquierda (LHS) que define las condiciones a cumplir.
- Una parte derecha (RHS) que define las acciones a realizar.

- Sintaxis:

```
(defrule nombre-regla"comentario"
      (condicion-1)(condicion-2)...
=>
      (accion-1)(accion-2)...)
```

2.2.3 ORIENTACIÓN A OBJETOS EN CLIPS

CLIPS define también una extensión orientada a objetos que complementa la capacidad de representar la estructura del conocimiento. Se puede considerar como una extensión del constructor *deftemplate* que pretende completar la posibilidad de usar *frames* como herramienta de representación Podemos definir clases como en los lenguajes orientados a objetos con *slots* y métodos.

CLIPS tiene definido un conjunto inicial de clases que organizan los tipos predefinidos de CLIPS estableciendo una jerarquía entre ellos. La sentencia que permite definir una clase es *defclass*, y para definir una clase hay que definir:

- Nombre de la clase.
- Lista de sus superclases (heredará de estas sus *slots* y métodos).
- Declaración de si es una clase abstracta o no.
- Si se permite que instancias de esta clase puedan vincularse a patrones en la LHS de una regla.
- Definición de los *slots* de la clase.

Por ejemplo:

```
(defclass ser-vivo
      (is-a USER)
      (role abstract)
      (pattern-match non-reactive)
(slot respira(default si)))
      (defclasspersona
      (is-a ser-vivo)
      (role concrete)
      (pattern-match reactive)
(slot nombre))
```

Para crear instancias de una clase de usa la sentencia make-instance por ejemplo:

```
(make-instance juan of persona(nombre"juan"))
```

Además se pueden crean conjuntos de instancias con la sentencia definstances, por ejemplo:

```
(definstances personas
     (juan of persona(nombre"juan"))
     (maria of persona(nombre"maria")))
```

2.3 SISTEMAS DE RECOMENDACIÓN

En la actualidad, existe una gran cantidad de sitios especializados en Internet, ofreciendo millones de productos y servicios para su consumo, convirtiéndose en un caos de información sin que el usuario pueda encontrar a veces lo que realmente desea. Los sistemas de recomendación surgen para solucionar este problema. Básicamente, los sistemas de recomendación reciben información del usuario acerca de productos o servicios en los que el usuario se encuentra interesado y le recomienda aquéllos que estén más cercanos a sus necesidades.

Para ello, se debe dotar a los sistemas de recomendación de unas ciertas técnicas que personalicen las recomendaciones más allá de la petición de productos por parte del usuario, por lo que aúna a diversos campos de investigación como son el marketing, interacción persona-ordenador, la minería de datos y la inteligencia artificial.

2.3.1 CLASIFICACION GENERAL

Dentro de los distintos sistemas de recomendación, se puede realizar una clasificación general y las diversas técnicas de recomendación.

A. **BASADOS EN CONTENIDO (*Content Based*)**: generan recomendaciones equiparando las preferencias del usuario expresadas de forma implícita o explícita) con las características utilizadas den la representación de los ítems ignorando la información relativa a otros usuarios. Se le recomendará al usuario un solo ítem similar al que el mismo usuario haya elegido anteriormente.

B. **FILTRADO COLABORATIVO (*Collaborative Filter*)**: también llamados sistemas de recomendación sociales. Son un tipo de sistemas de recomendación para los que se crea un conjunto de usuarios que tienen características o gustos similares entre sí. Si a un conjunto del grupo de usuarios les gusta un

determinado ítem, es de suponer que ese mismo ítem guste al resto de usuarios de ese grupo, es decir, se le recomendará un ítem que a las personas con los mismos gustos y preferencias les haya gustado en el pasado.

C. **RECOMENDACIÓN BASADA EN FACTORES ECONOMICOS**: es un sistema que busca las relaciones entre diferentes conceptos para así proporcionar una medida equitativa de recomendación, por ejemplo, la relación entre calidad y precio. Aunque es un sistema utilizado instintivamente por las personas en la vida común, su uso en sistemas de recomendación es muy marginal.

D. **RECOMENDACIONES DEMOGRÁFICAS (*Demographic*)**: clasifican a los usuarios de acuerdo a su perfil y hacen las recomendaciones basándose en clases demográficas. Son recomendaciones similares a las recomendaciones basadas en contenidos, salvo que las similitudes se calculan a partir de la utilización de información en lugar de la valoración de los ítems.

E. **RECOMENDACIONES BASADAS EN EL CONOCIMIENTO (*Knowledge based*)**: las sugerencias de los ítems se basan en inferencias sobre las necesidades de los usuarios y sus preferencias. Para ello, se utiliza conocimiento en donde se tiene la información sobre como un ítem específico responde a una necesidad particular del usuario, y por lo tanto, la razón sobre la relación entre la necesidad y una posible recomendación.

F. **RECOMENDACIONES BASADAS EN LA UTILIDAD (*Utility based*)**: son sistemas de recomendación que crean una función de utilidad para cada ítem la cual interviene directamente en el proceso de recomendación. La ventaja es que permite evaluar elementos no atribuibles al producto o ítem en sí. Existen otros aspectos, como la fiabilidad de un proveedor o la disponibilidad de un ítem, que son los que estarían representados en la función de utilidad.

G. **SISTEMAS DE RECOMENDACIONES SEMÁNTICAS**: mejoran y enriquecen la representación de la información mediante la aplicación de tecnologías de Web Semántica. Se pueden clasificar en tres tipos:

- Sistemas basados en ontologías o esquemas de conceptos: Las ontologías son utilizadas para representar la información o modelado de los ítems así como para la modelación de usuarios o perfiles de usuario.

- Sistemas adaptables al contexto: Toman en consideración diferentes factores del usuario tales como temporales, de lugar, nivel de experiencia, dispositivo que se está utilizando en el momento de recibir la recomendación, etc., para inferir el contexto en que se encuentra el usuario y adaptar las recomendación a esas circunstancias.
- Sistemas basados en redes de confianza: Añaden filtros de información adicionales a los sistemas adaptables al contexto.

H. **SISTEMAS DE RECOMENDACIÓN HÍBRIDOS (*Hybrid Recommender Systems*)**: combinan diversos tipos de sistemas de recomendación. El sistema híbrido más común utiliza las recomendaciones basadas en el contenido y además, el filtrado colaborativo. Como consecuencia, se atenúan las desventajas de cada una de ellos y se combinan los beneficios de las recomendaciones de ambos métodos.

2.3.2 PROBLEMAS GENERALES

Para todos los sistemas de recomendación descritos, existen una serie de problemas que son de ámbitos generales e independientes de los sistemas de recomendación que se utilicen.

- **Carencia de información (*Lack of data*)**: los sistemas de recomendación necesitan de mucha información para hacer efectivas sus recomendaciones. Así pues, para el funcionamiento de los algoritmos de recomendaciones, es decir, productos y usuarios. No siempre es necesario saber de ambos componentes, pero sí como mínimo de uno de ellos. Cuanta más información, más precisa será la recomendación. Este aspecto evoca al concepto de economía de escala, es decir, cuanto más grande sea el conjunto de usuarios o ítems con información, más probabilidades habrá de encontrar coincidencias entre perfiles o gustos de un usuario en concreto.

- **Información cambiante (*Changing data*)**: Hace referencia a los contenidos que su uso, modo o costumbre, están en alza durante algún tiempo, o en una determinada región, es decir, productos "de moda". Estos productos representan preferencias de usuarios en espacios de tiempo muy puntuales. El problema de este tipo de ítems radica en que cuando un determinado ítem está "de moda", recibe gran cantidad de valoraciones muy positivas. Por otro lado, cuando se termina el período de tiempo en que el ítem está "de moda",

los usuarios no desean recomendaciones del ítem; no obstante, el sistema de recomendación los sigue recomendado ya que tiene una gran cantidad de votos positivos del ítem (que fueron realizados en el periodo de tiempo en que el ítem estaba "de moda"). Este es un problema habitual en el mundo de la ropa. Por ejemplo, las tendencias de ropa de un año a otro varían completamente. En una temporada se pueden llevar los pantalones de pitillo, mientras que para la temporada siguiente se llevan los pantalones acampanados.

- **Cambio de preferencias de usuario (*Changing user preferences*)**: Usualmente los usuarios buscan recomendaciones para ellos mismos, pero eventualmente, estos mismos usuarios pueden buscar recomendaciones para otros usuarios de diferente perfil. Por ejemplo, un usuario determinado "A", del cual se tiene establecido un perfil con sus preferencias y gustos, le puede interesar buscar un cierto ítem para una persona "B" (por ejemplo para hacer un regalo de cumpleaños) con unas preferencias y gustos totalmente diferentes a los suyos.

- **Ítems impredecibles (*Unpredictable items*)**: Existen una serie de ítems que es difícil recomendarlos ya que la reacción del usuario hacia ellos puede ser diversa e impredecible. Por ejemplo, éste es el caso de la película "*Napoleon Dynamite*". Para la gente, este ítem tiende a ser o muy buena recomendación o muy mala, pero no existe un término medio.

- **Voto pronto y a menudo (*Vote early and often*)**: Según comenta Resnick y Varian si cualquiera puede hacer recomendaciones, los propietarios de un ítem pueden hacer recomendaciones positivas de ese ítem y recomendaciones negativas para los ítems de sus competidores. Este aspecto se puede considerar como un ataque a los sistemas de recomendación ya que se intenta sesgar las recomendaciones de cierto(s) ítem(s).

- **Complejidad computacional (*Computer complexity*)**: Los sistemas de recomendación tienen una mayor exactitud de sus predicciones cuando cuentan con una mayor cantidad de información disponible. No obstante, a mayor cantidad de información en el sistema, más costosos (en tiempo) son los cálculos para obtener los resultados. Una de las formas más habituales para reducir estos cálculos es el proceso *off-line* de los datos, es decir, en el momento de realizar una recomendación, ya se tienen datos que han sido calculados posteriormente (de forma *off-line*). Otra manera de abordar el problema consiste en la creación de grupos o "*clusters*" de elementos parecidos de tal forma que los elementos similares se agrupan y se computan como si fueran el mismo.

2.3.3 SISTEMAS BASADOS EN EL CONTENIDO

Los sistemas de recomendación basados en el contenido generan recomendaciones teniendo en cuenta la valoración que el usuario ha hecho de los ítems en el pasado. Así pues, el sistema tiene en cuenta las valoraciones que el usuario realiza de los ítems y las utiliza para recomendar nuevos ítems que tienen características similares a los ítems evaluados positivamente por el usuario.

Para poder inferir sobre el conocimiento (atributos o cualidades de los ítems) y realizar recomendaciones, es necesario almacenar de un modo u otro las características de cada ítem. Por lo general, este tipo de sistemas se suelen basar en el uso de una estructura bien definida para la anotación de las características y los valores de los correspondientes ítems. Por ejemplo, en un sistema de recomendación de viajes de vacaciones una posible estructura para representar las características de los viajes podría constar de campos como: precio, duración, localización, modo de transporte, tipo de alojamiento, etc. No obstante, no siempre es posible contar con una estructura de características y valores bien definida (por ejemplo en el caso de una crítica de cine o un artículo sobre un restaurante nuevo) por lo que se han propuesto diferentes algoritmos para analizar el contenido de los documentos (sin estructura o de texto plano) y encontrar regularidades en el contenido del documento que pueden servir como base para realizar recomendaciones.

El perfil del usuario es utilizado en los sistemas de recomendación basados en el contenido para almacenar información referente al usuario. Esta información puede ser proporcionada directamente por el usuario de forma explícita (por ejemplo, mediante un formulario) o deducida por el sistema (por ejemplo, mediante el feedback de los ítems que el usuario proporciona al sistema). El tipo de información que se puede contener en el perfil del usuario es:

- **Preferencias del usuario**: Contiene una relación de los tipos de ítems en que el usuario está interesado (o que no está interesado).

- **Historial del usuario**: Se almacenan las interacciones que el usuario ha tenido con el sistema. Por ejemplo, la valoración de un ítem. Dicho aspecto es importante ya que el sistema puede utilizarlo para aprender gustos y preferencias del usuario. La obtención de datos es necesaria en el sistema de recomendaciones basada en el

contenido pero no sirve de nada sin un mecanismo o algoritmo que los procese y extraiga conclusiones o, en este caso, recomendaciones. Estos algoritmos crean una función que modela los intereses de los usuarios. Por consiguiente, dado un modelo de usuario y un ítem nuevo, la función predice si el ítem será del agrado del usuario. A continuación se presentan diferentes formas de inferir en el conocimiento almacenado en el sistema.

- **Personalización del usuario**: El sistema proporciona una interfaz donde el usuario puede aportar las preferencias de sus gustos. Una vez que el usuario ha proporcionado sus preferencias, el sistema simplemente busca los elementos que encajan con los parámetros indicados por el usuario. El método implica que el usuario debe implicarse y rellenar un formulario o cuestionario para que el sistema pueda realizar las recomendaciones. Además, el método no provee ninguna forma para determinar el orden en que se presentan los resultados obtenidos por el sistema.

- **Basadas en reglas (*Rule based*)**: El sistema tiene una regla para recomendar productos basados en su historial. Por ejemplo, si el usuario ha adquirido los capítulos dos y tres de una determinada serie, el sistema recomendaría el capítulo cuatro de esa serie.

- **Reglas de asociación (*Associative rules*)**: Se intenta descubrir relaciones entre los ítems, que luego pueden ser utilizadas para recomendación. Los ítems se comparan en base al comportamiento pasado de los usuarios respecto a ellos. Por ejemplo, la compra de cereales puede relacionarse con la compra de leche.

- **Árboles de decisión (*Decision tree*)**: Los árboles de decisión se basan en una estructura en forma de árbol la cual reparte u organiza la información a lo largo de ésta. Las tres partes de que consta la misma son: nodos (contienen atributos), arcos (contienen valores posibles del nodo padre) y hojas (nodos que clasifican el ejemplo). Para elaborar recomendaciones, los algoritmos parten del nodo padre y evalúan los atributos de éste para seleccionar un arco. Este paso se repite iterativamente hasta alcanzar una hoja, la cual, es el resultado del algoritmo que identifica las recomendaciones posibles a tener en cuenta.

- **Método del vecino más cercano**: Permite clasificar un nuevo dato basándose en observaciones conocidas o pasadas. En el contexto de sistemas de recomendación basados en el contenido, se usa para determinar valores de un ítem sin valorar. Para ello, el nuevo ítem, que carece de valoración, se compara (mediante el uso de los atributos del ítem) con los otros ítems ya valorados. Mediante una función de similitud, se obtienen los vecinos más cercanos al nuevo ítem. El uso de las

etiquetas de los vecinos más cercanos, se utiliza para obtener o derivar una(s) etiqueta(s) para el nuevo ítem. Diferentes funciones son utilizadas para establecer el vecino más cercano dependiendo del tipo o estructura de datos con la que se trabaja. Las más usadas son la distancia euclidiana y el modelo de espacio vectorial.

- ***Feedback* relevante**: Método que propone obtener información sobre los ítems recomendados al usuario cuando éste ha realizado una búsqueda. Por ejemplo, páginas Web que ofrecen ayuda o soporte de sus programas, piden al usuario que valore la información aportada por el sistema conforme a lo que estaba buscando. Mediante la apreciación del usuario, el sistema conoce qué información ha sido útil para la búsqueda que se ha realizado y la tiene en cuenta para posteriores búsquedas, iguales o relacionadas, a la valorada por el usuario.

- ***Clustering***: Dado que muchas veces es una tarea costosa tanto en tiempo de cálculo como de requisitos, el obtener una recomendación para un individuo en concreto, una alternativa es agrupar usuarios en categorías en base a comportamientos pasados de otros consumidores. Cada grupo o *cluster* tiene preferencias que son típicas en base a sus miembros. Los usuarios dentro de cada grupo recibirán recomendaciones calculadas a nivel grupal. La ventaja del *clustering* es que trabaja sobre los datos agregados, es decir, más eficientes. Esta técnica se suele usar como un primer paso para reducir las búsquedas del filtrado colaborativo. Por otro lado las recomendaciones (a nivel grupal) son menos relevantes o precisas que en el filtrado colaborativo (a nivel individual).

- **Redes neuronales artificiales (Artificial neural network):** Las redes neuronales simulan las propiedades observadas en los sistemas neuronales de animales a través de modelos matemáticos recreados mediante mecanismos artificiales. El objetivo es conseguir que las máquinas (o algoritmos) den respuestas similares a las que es capaz de dar el cerebro que se caracterizan por su generalización y su robustez. Las redes neuronales están compuestas de muchos elementos sencillos que operan en paralelo.

- **Bayesiano naive (*Naïve Bayes*):** Método de clasificación probabilístico. Se utiliza para clasificar un nuevo ítem dentro de las preferencias del usuario. Por ejemplo, dado un nuevo ítem y un conjunto de usuarios, se calcula la probabilidad de que el ítem tenga cabida dentro de cada grupo de usuarios dependiendo de sus preferencias. Es un método importante ya que no sólo ofrece un análisis cualitativo de los atributos y valores que pueden intervenir en el problema, sino que además da a entender la importancia cuantitativa de esos atributos.

2.3.3.1 PROBLEMAS GENERALES

Los sistemas de recomendación basados en el contenido presentan los siguientes inconvenientes:

- Limitación por el análisis del contenido: El sistema se ve limitado por las propiedades que están explícitamente asociadas al ítem. Por consiguiente, es necesario tener las suficientes propiedades de cada objeto para un correcto funcionamiento.
- Sobre-especialización: La sobre-especialización aparece cuando el sistema sólo muestra al usuario ítems similares a los que ya ha visto antes. Por ejemplo, a una persona que nunca haya tenido una experiencia con restaurantes de comida japonesa, nunca se le recomendará un restaurante japonés.
- Un usuario nuevo: Un usuario tiene que haber valorado algunos ítems antes de que el sistema pueda saber de sus gustos y preferencias para poderle recomendar.

2.3.3.2 TENDENCIAS GENERALES

Los sistemas de recomendación basados en el contenido no son una solución en sí misma al problema de las recomendaciones, sino más bien sirven de ayuda o complemento a otros sistemas de recomendación, es habitual la combinación de los sistemas de recomendación basados en el contenido con los sistemas de filtrado colaborativo. Existe una tendencia a generar atributos o meta-información sobre los ítems de forma automática o semi-automática dado la creciente cantidad de información e ítems que hay hoy en día.

2.3.4 SISTEMAS DE FILTRADO COLABORATIVO

Los sistemas de filtrado colaborativo son sistemas de recomendación que utilizan la información que los usuarios aportan sobre los ítems para realizar las recomendaciones. Así pues, tratan de predecir las valoraciones de un (varios) ítem(s) para un usuario en particular (llamado usuario activo) mediante el uso de ítems valorados por otros usuarios, es decir, teniendo en cuenta la opinión de los otros usuarios. Como consecuencia, estos sistemas se basan en las tendencias marcadas por

grupos de personas con los mismos gustos o preferencias que el usuario activo para realizar las recomendaciones o sugerencias.

Los sistemas de filtrado colaborativo, por lo general, se adaptan al siguiente funcionamiento:

- Los usuarios elaboran valoraciones sobre ítems.
- Se analizan las valoraciones realizadas por los usuarios con tal de establecer grupos o vecinos cercanos de usuarios con preferencias similares.
- Una vez obtenidos los grupos de vecinos para un usuario, se realizan las recomendaciones al usuario activo teniendo en cuenta sus vecinos más cercanos y sus correspondientes valoraciones.

2.3.4.1 HISTORIA

Los sistemas de recomendación basados en el filtrado colaborativo aparecen después de los sistemas de recomendación basados en el contenido. En un principio los sistemas de recomendación basados en el contenido fueron de gran utilidad para una primera aproximación de filtrado de información, pero presentaban un inconveniente: No podían medir la calidad de las recomendaciones que realizaban. Frente a este problema, a principios de los noventa aparecieron dos posibles alternativas:

1. Esperar que técnicas de inteligencia artificial mejorasen la clasificación automática de los documentos o ítems.
2. Introducir la opinión de las personas en el proceso de las recomendaciones.

Con la adopción del punto 1, se mejoraban los algoritmos de recomendación basados en el contenido, mientras que el desarrollo del punto “2” produjo la creación de sistemas de recomendación social o de filtrado colaborativo.

2.3.4.2 CLASIFICACIÓN

Los sistemas de filtrado se clasifican en:

- Basados en la memoria o heurísticos: Son algoritmos que realizan sus predicciones teniendo en cuenta todos los elementos evaluados, es decir, utilizan toda la información disponible en el sistema.

- Basado en el modelo: Son un tipo de sistemas que usa un conjunto de valoraciones para crear un "modelo". Este modelo es utilizado a posteriori para crear predicciones de valoraciones.

El principal problema de los métodos heurísticos es la escalabilidad debido a la gran cantidad de información a procesar. Como consecuencia al problema de la escalabilidad, se utilizan técnicas para reducir el tiempo de proceso de la información como es el cálculo *off-line* de diferentes variables o constantes. Por otra parte, los algoritmos basados en el modelo utilizan conceptos probabilísticos que simulan la información presente en el sistema, pero que no es utilizada directamente. A continuación se describen diferentes técnicas usadas en los dos sistemas para alcanzar su objetivo.

a) BASADA EN LA MEMORIA

La predicción de votos de usuario activo en los sistemas de filtrado colaborativo basados en memoria se fundamenta en la medida de un conjunto de pesos calculado mediante el uso de la base de datos de votaciones. Los pesos pueden representar distancia, correlación o similitud entre el usuario activo y los diferentes usuarios de la base de datos. La distinción entre los diferentes sistemas de filtrado colaborativo basado en memoria viene determina por la forma de calcular estos pesos:

- Voto por defecto (*default voting*): Se aplica en el caso de los algoritmos de correlación en donde existen pocos votos, tanto por parte del usuario activo, como de los otros usuarios. Se ha demostrado empíricamente que la precisión de los resultados (en los algoritmos de correlación) es mejorada si se asumen valoraciones por defecto en los ítems no valorados.

- Frecuencia inversa del usuario (*inverse user frequency*): La idea es aplicar el mismo principio que "inverse document frequency" a los usuarios de la base de datos. Es decir, reducir los pesos para las valoraciones de los ítems valorados más frecuentemente con el fin de identificar los ítems más relevantes. El sentido de este método es que los ítems de gusto general no capturen la similitud entre usuarios.

- Amplificación de casos (*case amplification*): Se trata de enfatizar los pesos que están más cerca los unos de los otros y al mismo tiempo castigar los pesos menores. Para ello, se aumentan de forma proporcional los pesos de las

valoraciones próximas y se les resta valor a las valoraciones que se encuentran alejadas.

b) BASADAS EN EL MODELO

Los algoritmos, basados en el modelo, tienen un enfoque probabilístico. Los diferentes algoritmos empleados son:

- Redes Bayesianas: Una red Bayesiana se basa en la creación de un grafo el cual relaciona los diferentes nodos de éste de forma probabilística. En el contexto del filtro colaborativo basado en el modelo, cada nodo corresponde a un ítem y su estado es determinado por la posibilidad de que ese nodo/ítem sea votado por los otros ítems del grafo. A este grafo se le aplican algoritmos para encontrar dependencias entre nodos y así poder obtener para cada ítem, un conjunto de ítems padres que corresponden a las mejores predicciones de sus votos.

- *Clustering*: Se trata el problema colaborativo como si fuera un problema de clasificación por lo que crea grupos de usuarios similares (o *cluster*) y en lugar de procesar usuarios, procesa *clusters*. Los algoritmos, en lugar de comparar el usuario activo con todos los usuarios del sistema, comparan el usuario activo con grupos de usuarios, es decir, los *clusteres*.

- Redes neuronales artificiales: Al igual que en los sistemas de recomendación basados en el contenido, las redes neuronales, también tienen cabida en los sistemas de recomendación de filtrado colaborativo.

- Redes basadas en grafos (*graph-theoretic approach*): La idea del método es la formación de un grafo dirigido cuyos nodos corresponden a los usuarios y a sus relaciones se les asigna un peso y dirección. Cuando un usuario predice a otro usuario, se crea una transformación lineal que es construida con las valoraciones "trasladadas" de un usuario a otro. De este modo, la predicción de recomendación de un ítem para un usuario se realiza mediante la media de pesos de las direcciones de las relaciones, antes mencionadas, de los diferentes usuarios.

- Popularidad del impacto de proximidad (*Proximity Impact Popularity*): El enfoque, propuesto por Hyung Jun Ahn, intenta solventar el problema del arranque en frío. La popularidad del impacto de proximidad es una medida

heurística que se basa en tres factores: (a) la proximidad: se refiere a la distancia entre las valoraciones y representa la diferencia aritmética entre dos valoraciones; (b) el impacto que representa la fuerza con la que las valoraciones son del agrado o desagrado de los usuarios y (c) la popularidad que incrementa el valor de las similitudes entre valoraciones que estén lejos de la media (de los ítems ya valorados).

2.3.4.3 PROBLEMAS

Los sistemas de filtrado colaborativo presentan los siguientes problemas:

- Nuevo usuario: Cuando un usuario llega al sistema, no es posible hacerle recomendaciones hasta que su perfil sea lo suficientemente completo para encontrarle a su grupo de vecinos cercanos. Existen diferentes técnicas que buscan paliar el problema intentando determinar el mejor ítem para los nuevos usuarios que estos deben valorar.

- Arranque en frío: El arranque en frío se suele dar en la etapa inicial de la utilización del sistema, cuando la mayoría de los ítems aún no han sido calificados y, por lo tanto, no son utilizados para los cálculos de similitud. El mismo problema puede manifestarse, aún cuando el sistema ha sido extensamente utilizado, cuando se agregan nuevos ítems o con aquellos ítems que representan gustos selectos de usuarios particulares.

- Problema de Matriz dispersa: Surge debido a que cada usuario sólo califica a un número relativamente reducido de ítems. Por tanto la matriz de calificaciones usuario-ítems es una matriz dispersa, con numerosos valores nulos. Como resultado de lo anterior, la probabilidad de encontrar perfiles similares es habitualmente baja.

- Confianza entre usuarios: En las recomendaciones del tipo colaborativo, se puede dar el caso de que existan usuarios mal intencionados que intentan corromper el sistema de recomendación. Para solucionarlo, la idea es que un usuario, además de puntuar los ítems del sistema, también exprese su confianza en las valoraciones que han hecho otros usuarios.

2.3.4.4 TENDENCIAS GENERALES

Al tratar con información de carácter personal, es inevitable entrar en temas relacionados con la seguridad y privacidad de la información. Los usuarios, de forma implícita, confían en que los sistemas de recomendación sólo utilizarán sus preferencias para realizar recomendaciones y que sus datos permanecerán en el anonimato. Según comenta Canny, en el futuro, los usuarios querrán obtener recomendaciones sobre diferentes aspectos de sus actividades diarias, como pueden ser, comidas en restaurantes, sesiones de cine, o visitas a lugares cercanos de su lugar de residencia. Aunque en este sentido la ley tiene presente aspectos de privacidad, se puede dar el caso de que empresas que se dedican al comercio electrónico recojan información sobre sus usuarios y hagan el esfuerzo de mantenerla de forma privada hasta el momento que por ejemplo, entran en banca rota y aprovechan para vender este tipo de información, de carácter privado, a terceros. Por ello, los usuarios deben poder tener control exclusivo a toda la información relacionada con él.

2.3.5 RECOMENDACIONES BASADAS EN EL CONOCIMIENTO

Los sistemas de recomendaciones tradicionales, como son los de filtrado colaborativo o basados en el contenido, se apoyan en perfiles de usuario y ciertas características de los ítems para realizar las recomendaciones. No obstante, estos sistemas no explorar en profundidad el conocimiento sobre el dominio de los ítems. Así pues, los sistemas de recomendaciones tradicionales son perfectamente válidos para procesos de recomendación de productos singulares como CDs musicales, libros o películas, pero no son útiles en dominios complejos.

Los sistemas de recomendación basados en el conocimiento pretenden profundizar en el conocimiento sobre los usuarios y los ítems - mayoritariamente de domino complejo- para la elaboración de recomendaciones que encajen de forma adecuada con los requisitos del usuario.

Los sistemas basados en el conocimiento hacen una representación explicita sobre el producto así como las oportunidades que lo envuelven, permitiendo:

- La recomendación de ítems que satisfagan ciertos requisitos de calidad.
- La explicación del porqué de la recomendación realizada. Este hecho tiene una implicación positiva en la confianza de los usuarios hacia las recomendaciones del sistema.
- El soporte a los usuarios cuando no se puede encontrar una solución o recomendación adecuada para el caso del usuario.

Por otra parte, los sistemas de recomendación basados en el conocimiento presentan las siguientes desventajas:

- Es necesaria una ingeniería del conocimiento.
- Las recomendaciones son del tipo estático, es decir, para los mismos casos, se realizan las mismas recomendaciones.

Un caso particular de las recomendaciones basadas en la utilidad son las recomendaciones basadas en la utilidad. Su objetivo es el de crear un valor de "utilidad" (beneficio, ventaja o interés) para los ítems a recomendar para un usuario en particular. En principio el valor de utilidad se basa en la utilización del conocimiento. Por ello, este tipo de sistemas hereda los beneficios de los sistemas de recomendación basados en el contenido como es la eliminación del problema del arranque en frío. No obstante, añaden la necesidad de crear una "*función de utilidad*", que permite obtener un valor de utilidad para cada ítem a recomendar el cual proporciona un baremo al usuario que refleja el grado de satisfacción del ítem con sus necesidades, gustos o preferencias.

2.3.5.1 FUNCIONAMIENTO

Los algoritmos que se usan en los sistemas basados en el conocimiento están fundados en el razonamiento basado en el caso. Se pueden distinguir tres tipos diferentes de conocimiento:

- Conocimiento del catálogo (*catalog knowledge*): Es el conocimiento que el sistema tiene sobre los ítems y sus características. Por ejemplo, el tipo de cocina "Thai" pertenece al grupo de cocina "Asiático".
- Conocimiento Funcional (*functional knowledge*): Es el conocimiento de cómo los ítems pueden coincidir con las necesidades de los usuarios. Por ejemplo, el sistema puede saber que para la necesidad de una cena "romántica", un restaurante apropiado seria uno "tranquilo con vistas al mar".

- Conocimiento del usuario (*user's knowledge*): El sistema necesita reunir información sobre las necesidades del usuario para poder encontrar los ítems que satisfagan sus necesidades.

2.3.5.2 CONCLUSIONES

Los sistemas de recomendación basados en el conocimiento incorporan soluciones validas para usuarios que requieren de una recomendación que se ajuste a unas necesidades concretas. Además, ya que presentan una ingeniería del conocimiento del dominio en el que trabajan, eliminan problemas generales de los sistemas de recomendación como son el arranque en frío o anomalías estáticas del mercado

Los sistemas basados en el conocimiento tienen una presencia relativamente marginal donde su uso se limita a dos situaciones muy concretas:

- En dominios muy precisos como son los entornos financieros. Las recomendaciones realizadas en este tipo de casos son muy ajustadas a unas necesidades y no son de carácter repetitivo. Por ejemplo, un usuario que pide un préstamo al banco para la compra de una casa, muy posiblemente no vuelva a pedir otro préstamo al banco al día siguiente.

- Como apoyo en fases tempranas de implementación de otros sistemas de recomendación, es decir, para eliminar el problema del arranque en frío de otros tipos de sistemas de recomendación. Por ejemplo, se quiere implementar un sistema basado en el filtro colaborativo, pero no se tiene suficiente información de los usuarios para eliminar el problema de arranque en frío. En este caso, se implementa el sistema de recomendación basado en el conocimiento hasta que el sistema tiene suficiente información sobre los usuarios para que no aparezca el problema del arranque en frío. En ese instante, es cuando se prescinde del sistema de recomendación basado en el conocimiento para pasar a utilizar el filtro colaborativo.

2.3.6 CONCLUSIONES A LOS SISTEMAS DE RECOMENDACIÓN

Cada uno de los diferentes tipos de sistemas de recomendación realiza un enfoque diferente para la realización de recomendaciones. Por ello, aparece la necesidad de estudio y comprensión del dominio o medio en que deben ser aplicados

o implementados para la maximización de los resultados (en forma de recomendaciones). Además, el estudio y comprensión del dominio de aplicación del sistema de recomendación debe comprender la necesidad de minimización de los problemas que acarrean los diferentes tipos de sistemas de recomendación. Una posible solución para la mitigación de los problemas asociados a cada uno de los diferentes tipos de sistemas de recomendación es la combinación de estos, es decir utilización de sistemas híbridos.

3. ANÁLISIS DE REQUISITOS

Para el desarrollo del proyecto, es conveniente realizar un análisis de requisitos para conocer las necesidades funcionales del proyecto. Primeramente, es básico diseñar la ontología necesaria para realizar el diagrama de clases para la estructura del código CLIPS, para la generación de las recomendaciones turísticas. Por otra parte, se debe plantear el diseño de la aplicación Web, con los elementos a incluir, la disposición de dichos elementos, la base de datos para gestionar la aplicación, con las actividades, generaciones y recomendaciones.

3.1 Diseño Ontología

En la fase de realización de la Ontología, se estudian los diversos rasgos a tener en cuenta. Las actividades o intereses más característicos son también tenidos en cuenta para dividirlas en diferentes grupos de actividades. Así, finalmente, se decide que los rasgos que debe tener nuestra ontología en la clase llamada PERSONA son Nombre, Edad, Discapacidades (motriz, visual y auditiva), los intereses en distintas actividades, el presupuesto del que dispone el usuario, la ciudad que desea visitar y en qué zona de la ciudad se encuentra y por último la disponibilidad horaria. Las distintas actividades con sus atributos se describen a continuación:

ACTIVIDAD
CLASE-PADRE:
ID: int
Nombre: string
Adaptado: Instancia a Minusvalía->Tipo
Edad mínima: int
Franja día: {Mañana, Comida, Tarde, Cena, Noche}
Días abierto: {lunes, martes, miércoles, jueves, viernes, sábado, domingo}
Precio: float
Grado de afición mínimo para ser recomendado: int (1..5)
Zona: Instancia a Barrio->Transporte

Museos
CLASE-PADRE: ACTIVIDAD

Centros Deportivos

CLASE-PADRE: ACTIVIDAD

Centros de Compras

CLASE-PADRE: ACTIVIDAD

Restaurantes

CLASE-PADRE: ACTIVIDAD

Teatros

CLASE-PADRE: ACTIVIDAD
Día del teatro: miércoles
Precio día del teatro: float

Cines

CLASE-PADRE: ACTIVIDAD
Día espectador: martes
Precio día espectador: float

Ocio Nocturno

CLASE-PADRE: ACTIVIDAD
Edad mínima: 18 (Restricción)

Minusvalía

Tipo: string

Barrio

Nombre: string
Trasportes: Instancia a Transporte -> Tipo

Transporte

Nombre Parada: string
Tipo: {metro, autobús}
Precio: float

Persona
Nombre: string
Discapacidad: Instancia a -> Minusvalía
Edad: integer
Presupuesto: float
Grado interés centro comercial: {1..5}
Grado interés centro cultural (museo): {1..5}
Grado interés centro cultural (cine): {1..5}
Grado interés centro cultural (teatro): {1..5}
Grado interés centro deportivo: {1..5}
Grado interés centro comercial: {1..5}
Grado interés ocio nocturno: {1..5}
Día disponible: {Lunes, Martes, Miércoles, Jueves, Viernes, Sábado, Domingo}
Franja del día disponible: {Mañana, Comida, Tarde, Cena, Noche}

Para la implementación de dicha ontología, se utiliza el programa *Protégé*. Es un *software* de la Universidad de Stanford. Una herramienta creada para el diseño y la implementación de ontologías que permite a través de la definición previa de la ontología en formato de diagrama de clases crear un sistema que permita responder las preguntas generadas en la ontología por ejemplo. En la siguiente imagen se muestra la definición de las clases.

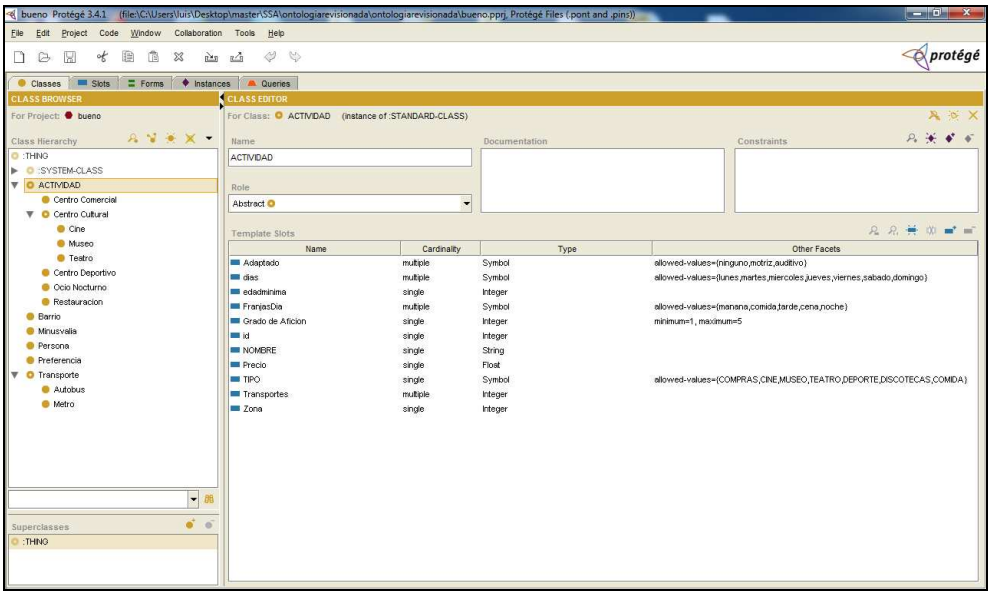


Ilustración 4: Diseño Ontología

La figura 5 muestra los diferentes atributos para cada clase de la ontología, donde se pueden insertar las instancias (o también llamados, objetos) de las clases.

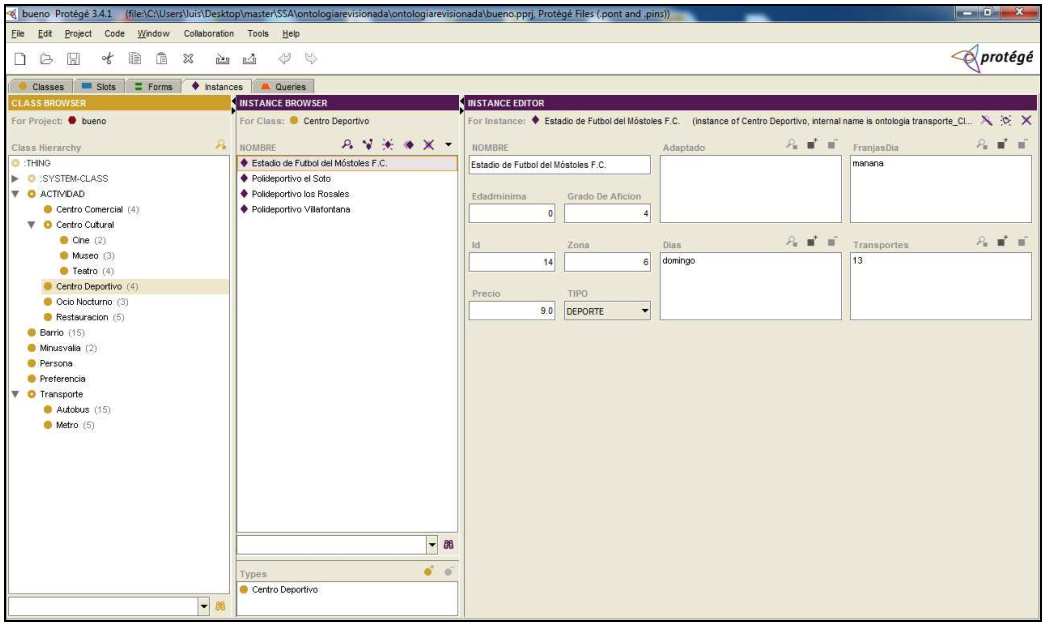


Ilustración 5: Introducción Instancias

Una vez realizado este proceso, se puede utilizar la opción de *Protégé* de transformar este diseño a lenguaje CLIPS para poder generar las distintas recomendaciones a través de este lenguaje de sistemas expertos.

Un ejemplo de clase en lenguaje CLIPS es:

```
(defclass ACTIVIDAD
        (is-a INITIAL-OBJECT)
        (role abstract)
        (pattern-match non-reactive)
        (slot Precio
                (type FLOAT)
                (create-accessor read-write))
        (slot ID
                (type INTEGER)
                (create-accessor read-write))
        (multislot Dias
                (type SYMBOL)
                (allowed-values lunes martes
miercoles jueves viernes sabado domingo)
                (create-accessor read-write))
        (slot Ciudad
                (type STRING)
                (create-accessor read-write))
        (slot GradoDeAficion
                (type INTEGER)
                (range 1 5)
                (create-accessor read-write))
        (slot Zona
                (type INTEGER)
                (create-accessor read-write))
        (slot EdadMinima
                (type INTEGER)
                (create-accessor read-write))
        (slot Nombre
                (type STRING)
                (create-accessor read-write))
```

```
(multislot FranjasDia
        (type SYMBOL)
        (allowed-values manana comida
tarde cena noche)
        (create-accessor read-write))
(slot Tipo
        (type STRING)
        (create-accessor read-write))
(multislot Transportes
        (type INTEGER)
        (create-accessor read-write))
(slot LEIDO
        (type SYMBOL)
        (allowed-values si no)
        (default no)
        (create-accessor read-write))
(multislot Adaptado
        (type SYMBOL)
        (allowed-values ninguno motriz
auditivo visual)
        (default ninguno)
        (create-accessor read-write)))
```

Otro ejemplo sobre el lenguaje CLIPS es el de una instancia, que se muestra a continuación:

```
([ontologia+transporte_Class20006] of  Cine
        (Adaptado ninguno auditivo)
        (DiaEspectador martes)
        (Dias jueves viernes sabado domingo)
        (EdadMinima 0)
        (FranjasDia manana tarde noche)
        (GradoDeAficion 2)
        (ID 10000006)
        (Nombre "Cine VOS")
        (Precio 5.0)
        (PrecioDiaEspectador 3.0)
        (Tipo CINE)
        (Transportes 12)
        (Zona 12))
```

3.2 Diseño Base de Datos

Para tener un mejor control y acceso a los datos, es necesario crear una base de datos para almacenar los diferentes datos sobre las actividades. Para el diseño de la base de datos, primeramente se debe estudiar qué elementos son necesarios guardar o tener en cuenta para la aplicación.

Las actividades son elementos básicos dentro de la aplicación, por lo que es vital la creación de una tabla en la que los diferentes campos o atributos son los definidos en la ontología para tener toda la información. Además, ya que vamos a disponer de distintas ciudades y barrios, es conveniente crear dichas tablas para realizar una correcta abstracción de la información.

Para guardar las generaciones realizadas a los usuarios, es necesario crear también otra tabla, donde se guarde información sobre la actividad recomendada a través de la generación, el día y la franja en la que ha sido recomendado y el precio de dicha actividad.

Por último, para realizar las recomendaciones en otros usuarios, se deben guardar diferentes datos como la actividad que está siendo vista, la actividad que puede ser recomendada por estar relacionada y un contador con el número de veces en el que se contabiliza el número de veces que a partir de la primera actividad ha sido seleccionada o visitada la segunda actividad.

Después de analizar las tablas que se deben de implementar, se realiza el diseño de la implementación con las diversas dependencias de claves ajenas entre las distintas tablas. Así el diseño se muestra en la siguiente imagen:

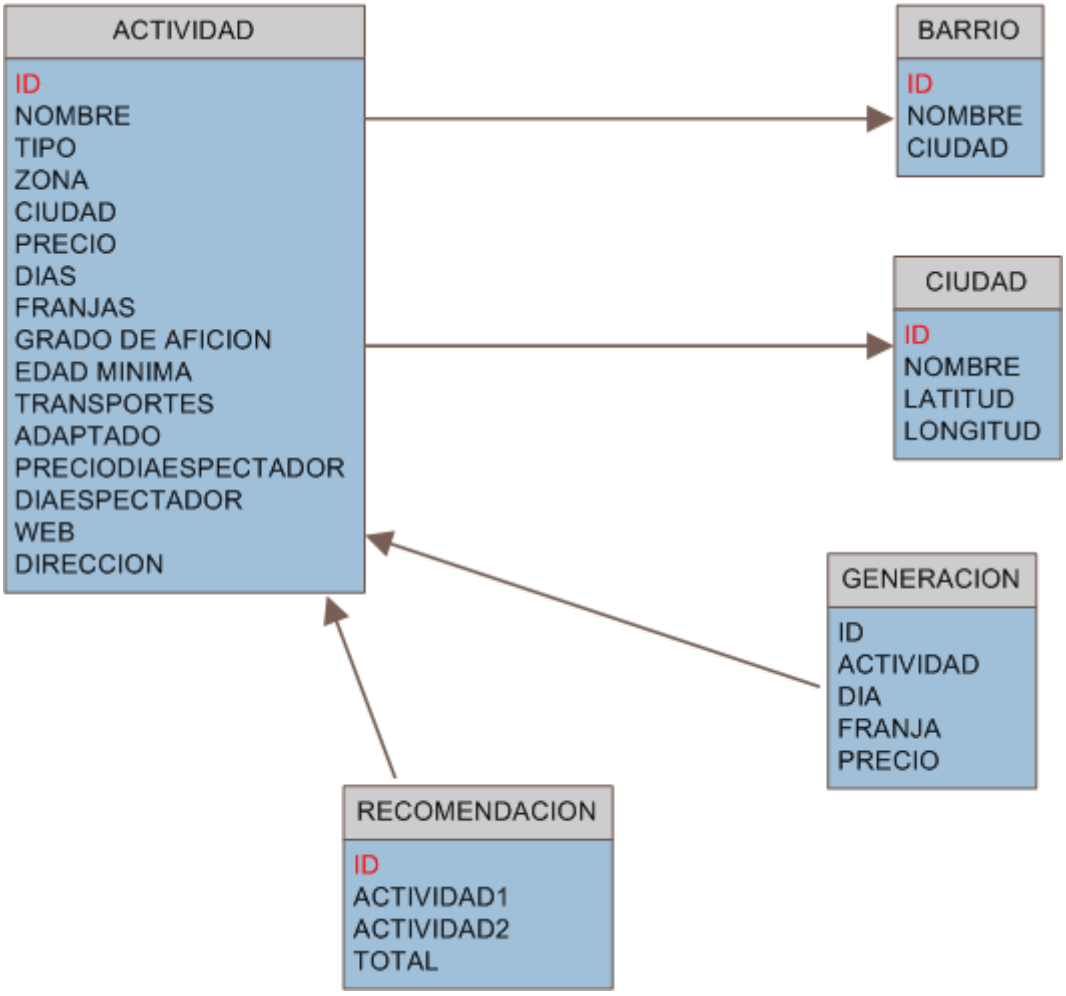


Ilustración 6: Diseño Base de Datos

Como se muestra en la figura 6, la tabla actividad utiliza las claves ajenas de Zona y Ciudad. Las tablas Recomendación y Generación utilizan la clave ajena actividad para la sus campos correspondientes, en el caso de Recomendación para 2 campos, ya que se utilizan dos actividades.

La base de datos que se elige es SQL SERVER 2008, ya que dispone de una gran interactividad e integración con ASP.NET 2.0 y Microsoft Visual Studio 2008.

3.3 Diseño Aplicación Web

En la parte del diseño de la aplicación Web, tan importante es crear un sitio robusto como usable para el usuario. También es importante crear una distribución de los distintos elementos acorde con las limitaciones tecnológicas (el caso de los tamaños de los monitores, el tamaño que pueda tener la página Web en sí) como el realizar un diseño de los elementos para que sea atractivo para el usuario y puedan cumplir con la función principal, la de poder realizar una recomendación.

Primeramente se analizan los requisitos principales y más básicos de la aplicación.

TAMAÑO DE LA PÁGINA	
Descripción	Especificación del tamaño de la página.
Explicación	La limitación en cuanto al tamaño de los monitores y la necesidad de que pueda estar presente la mayor cantidad de información posible de un "vistazo".
Valores	Ancho: 850 px.

COLORES	
Descripción	Especificación de los colores a utilizar en la página
Explicación	Para una correcta visualización, es necesario crear un buen contraste de colores para que la lectura sea correcta y fácil para el usuario.
Valores	Fondo claro y letra oscura.

FUENTE	
Descripción	Especificación de la fuente a utilizar en los textos.
Explicación	Para una correcta visualización, es necesario utilizar una fuente que sea bien legible.
Valores	16px

TAMAÑO FUENTE	
Descripción	Especificación del tamaño de fuente a utilizar en los textos.
Explicación	Para una correcta visualización, es necesario utilizar un tamaño de letra lo suficientemente grande para ser leído correctamente, pero no tan grande como para resultar enorme.
Valores	Tamaño de letra de 12 puntos.

SERVIDOR	
Descripción	Especificación del servidor que debe hospedar la página.
Explicación	Para poder utilizar la tecnología .NET se debe utilizar un servidor Microsoft para albergar la aplicación.
Valores	IIS7 (Internet Information Services) de Microsoft.

NAVEGADORES	
Descripción	Especificación de los navegadores en los que debe verse la página.
Explicación	Para una correcta visualización, es necesario probar y validar las páginas en los distintos navegadores que existen en el mercado, para ello, se hace una selección de los navegadores más usados por los usuarios de Internet.
Valores	Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Safari, Opera.

3.3.1 ELEMENTOS

Dado que para la recomendación se deben de facilitar diferentes datos o preferencias, es conveniente identificar los elementos que son necesarios para la recomendación. Primero la de introducción de datos, con 3 apartados bien diferenciados: datos del usuario, aficiones y disponibilidad horaria.

Para el apartado sobre los datos del usuario, es necesario habilitar una zona donde el usuario introduzca los datos imprescindibles para realizar a recomendación personalizada, como son:

- La edad, necesaria para determinadas actividades.
- El presupuesto del que dispone o que desea gastarse como máximo el usuario.
- La ciudad donde se encuentra o que desea visitar.
- El barrio donde se sitúa.
- Discapacidades que el usuario sufre (motriz, auditiva y/o visual).

Para el apartado de Aficiones, el usuario debe valorar de 1 a 5 las distintas actividades disponibles (cine, teatro, ocio nocturno, museo, compras, deporte, parques y gastronomía).

Por último, es necesario que el usuario identifique los días y franjas horarias en las que está disponible y desea realizar actividades.

Después de realizar la introducción de los datos, al usuario se le redirigirá a otra página dónde se le mostrará las recomendaciones. Para esta página, será necesario también, mostrar la información dividida en diferente bloques o elementos.

Por una parte, se debe mostrar un calendario, en el que el usuario pueda ver de forma sencilla que días y en que franjas ha recibido la recomendación.

Otro elemento imprescindible es mostrar a los usuarios las características de la actividad elegida para que el usuario disponga de información. Para ello se elige que la información a mostrar sea:

- El nombre de la actividad para que el usuario la conozca
- La dirección donde se encuentra
- El tipo de actividad que es, cine, deporte, etc.
- El precio de realizar dicha actividad
- Web de la actividad en caso de que disponga de ella
- Zona en la que se encuentra
- Transporte para poder realizar dicha actividad, autobús, metro o ambas.

Para dar una mayor información al usuario, se incluye un mapa sobre la situación de la actividad.

Además, se le informará al usuario de otras actividades similares que le puede interesar según la experiencia de otros usuarios con la navegación y recomendación que han realizado anteriormente.

En lo que respecta a la distribución de los diferentes elementos, se realiza un análisis con diferentes pruebas para elegir el más adecuado con los requisitos y con un aspecto atractivo para el usuario.

3.3.2 PRUEBAS DE DISEÑO

Para elegir la distribución, se realizan varias pruebas de la página en la que el usuario introducirá sus datos para realizar la recomendación.

En la primera prueba (ver ilustración 7), en la parte izquierda superior se muestra los datos personales, debajo de esta parte las aficiones del usuario y en la parte derecha el calendario para que el usuario elija su disponibilidad horaria.

En la segunda prueba (ver ilustración 8), en la parte izquierda superior se muestra los datos personales, en la parte derecha superior las aficiones de los usuarios y en la parte inferior el calendario donde el usuario pueda indicar su disponibilidad horaria.

Se elige la prueba de diseño 2, ya que supone una mejor distribución de los elementos a mostrar para el usuario.

Ilustración 7: Página 1 Prueba 1

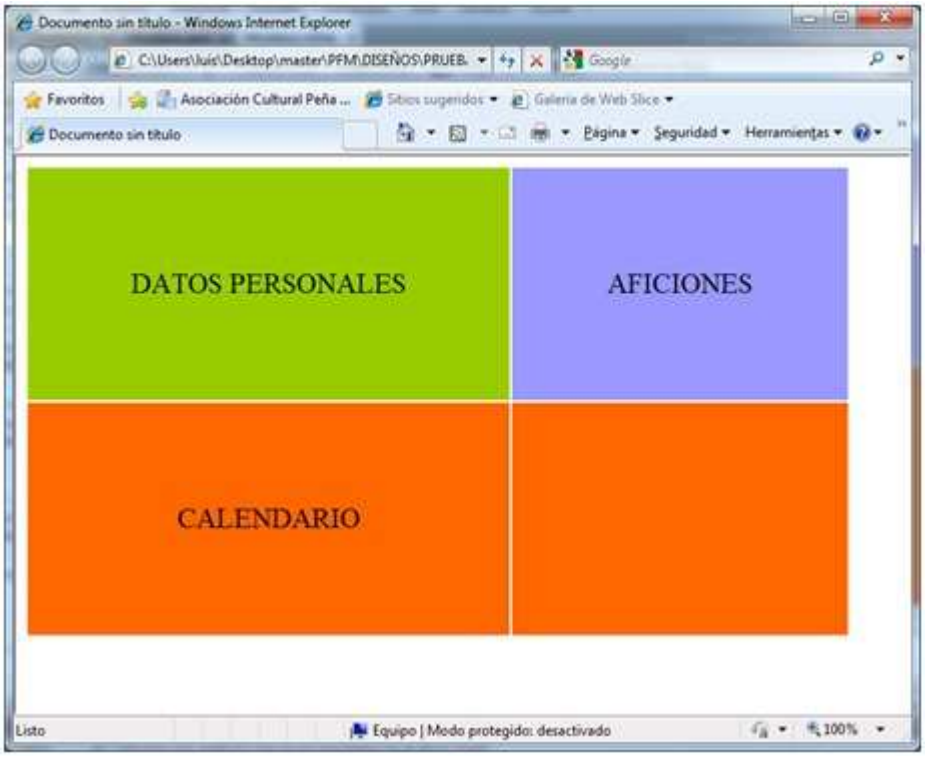


Ilustración 8: Página 1 Prueba 2

Para la página en la que se muestran los resultados de recomendación se realizan las siguientes pruebas:

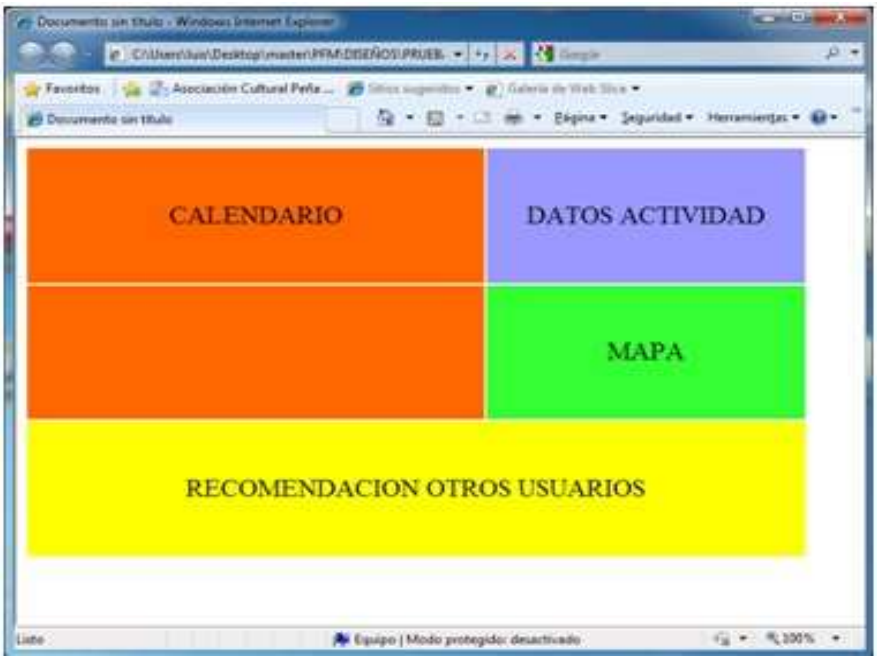


Ilustración 9: Página 2 Prueba1

En esta prueba, el elemento del calendario con las recomendaciones efectuadas por el sistema, se coloca en la parte izquierda superior. En la parte derecha se coloca los datos de la actividad elegida por el usuario y justo debajo el mapa de situación de la actividad. En la parte inferior de la página, se sitúan las recomendaciones de otros usuarios.

Ilustración 10: Página 2 Prueba 2

En esta prueba, la colocación de los diferentes elementos es muy similar a la de la prueba 1, salvo intercambiado los elementos mapa y datos actividad de lugar.

Ilustración 11: Página 2 Prueba 3

En esta prueba, se coloca el elemento calendario en la parte superior de la página. Debajo de él, a la izquierda se muestra los datos de la actividad seleccionada y a la derecha el mapa de situación. El elemento de recomendación de otros usuarios estaría en la parte inferior de la página.

Ilustración 12: Página 2 Prueba 4

En esta prueba, la colocación de los diferentes elementos es muy similar a la prueba 3, salvo intercambiado de lugar los elementos mapa y datos actividad.

Finalmente se elige el diseño número 3, ya que es una disposición de los elementos mucho más natural. Primero se muestra el calendario con las recomendaciones realizadas, el usuario debe pulsar sobre uno de los botones para que se le muestre la información, por lo que parece más lógico mostrarse en la parte inferior. Además es también más lógico la izquierda, ya que un usuario mientras está navegando suele ver primero los elementos que tiene en la parte superior-izquierda y de arriba abajo en diagonal.

Además, también parece más lógico mostrar las recomendaciones de otros usuarios en la parte inferior de la página, para no distraer demasiado al usuario, ya que la información que desea ver primeramente es la de la actividad que le ha sido asignada, y posteriormente si le interesa, ver otras recomendaciones.

4. IMPLEMENTACIÓN

En el siguiente apartado, se detalla la implementación de la aplicación Web, la implementación de Ontología con sus reglas en CLIPS y la aplicación en sí, con ASP.NET 2.0.

4.1 CLIPS

Para la ejecución del proyecto, una parte fundamental es la implementación de las reglas en el sistema experto CLIPS para la generación de las diferentes rutas a través de técnicas de Inteligencia Artificial. Los pasos a seguir para la generación de las rutas se muestra en el siguiente grafico:

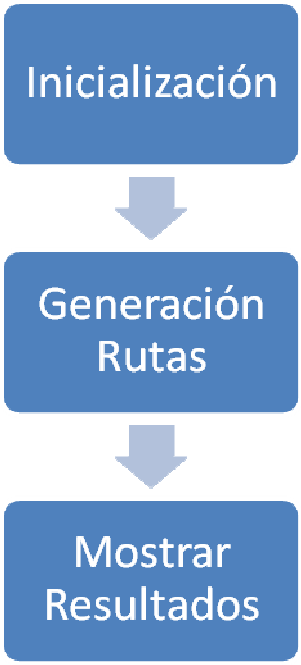


Ilustración 13: Pasos de la Aplicación

En la fase de inicialización se crean diferentes hechos para llevar un control sobre la ejecución de las reglas y se añade la aletoriedad a la hora de ejecutar reglas con la misma prioridad con la sentencia (*set-strategy random*).

En la fase de Generación de Rutas se definen todas las reglas en las que se asignan las recomendaciones para las distintas actividades, teniendo en cuenta el grado de afición, el presupuesto del que se dispone, la disponibilidad horaria y la edad.

```
(defrule asignar-cine

        (declare (salience 70))

        ?persona <- (object(is-a Persona)(Edad ?e)(Presupuesto ?presup)(Total ?tot)(GradoCine ?gr1)(Discapacidades $?w ?d1 $?w2))

        ?actividad <- (object(is-a ACTIVIDAD)(ID ?id)(Tipo CINE)(FranjasDia $?fr1 ?lib $?fr2)(Dias $?da ?dia $?dd)(LEIDO no)(Precio ?pre)(EdadMinima ?em)(GradoDeAficion ?gr2)(Adaptado $?s ?d1 $?a))

        ?preferencia <- (object (is-a Preferencia) (Dia ?dia)(FranjasDia ?lib)(Actividad 0))

        (contadores (preferencia ?gr1))

        (test (> ?presup (+ ?pre ?tot)))

        (test (>= ?e ?em))

        (test (>= ?gr1 ?gr2))

=>

        (modify-instance ?actividad (LEIDO si))

        (modify-instance ?persona (Total (+ ?tot ?pre)))

        (modify-instance ?preferencia(Actividad ?id))

        (make-instance of Asignacion (Tipo CINE)(ID ?id)(Precio ?pre)(FranjasDia ?lib)(Dia ?dia)))
```

En este ejemplo, la regla asigna al usuario la actividad CINE, se declara una cierta prioridad a la regla (ésta es una manera de controlar el transcurso de la ejecución del código).

Se necesita una instancia de persona, con una cierta edad ?e, un presupuesto ?presup, y discapacidades (puede no tener ninguna). Otra instancia de Actividad en la que debe estar disponible en una determinada franja y un día determinado, un precio y una adaptación a discapacidades (puede no estar adaptada a ninguna). Por ultima una instancia de las preferencias del usuario, con el día y la franja, que deberá ser en la que la actividad está disponible.

Se comprueba que el asignar esta actividad no suponga sobrepasar el presupuesto establecido, que la actividad pueda realizarla el usuario en lo que respecta a su edad y que tenga un grado de afición mínimo para ser recomendado.

Si cumple con esas reglas, se modifican las instancias de actividad y de la persona para actualizarlas y se crearía la instancia correspondiente de asignación para guardar la recomendación realizada. Esta regla se repetiría con todas las demás actividades.

Por último, para terminar con el proceso de CLIPS, se crean las reglas para imprimir en un fichero .txt la recomendación realizada, que luego será leída por la aplicación Web. En el fichero se imprimirá el día y franja de realización de la actividad, el id de la actividad, el precio y el tipo de actividad.

4.2 APLICACIÓN WEB

En este apartado, se explica la implementación de la aplicación, explicando su ciclo de vida correspondiente y la interacción con otras aplicaciones, por ejemplo CLIPS.

En la página de la introducción de datos, existirán diferentes Textbox donde el usuario introducirá su nombre, edad y presupuesto. También existen 3 checkbox en los que el usuario indica si padece alguna discapacidad (Motriz, Auditiva y/o Visual). Además de elegir la ciudad a través de un Listbox. La información del Listbox procede de la base de datos a través de un SqlDataSource de ASP.NET. A partir de la elección de la ciudad se crea otro Listbox con los barrios, que también cogen la información de la base de datos a través de otro SqlDataSource. Por último, se incluye un calendario, formado por checkboxs, cada uno correspondiente a un día y franja.

Una vez introducidos estos datos, se realizan diferentes validaciones para que se pueda realizar la recomendación. Para estas validaciones, se utilizan distintos elementos de ASP.NET 2.0 y que ayudan a hacer más fácil las validaciones, sin tener que implementarlas en lenguaje JavaScript. Se utilizan los elementos RequiredFieldValidator, RangeValidator, CustomValidator y ValidationSummary.

RequiredFieldValidator: elemento para asegurar que un campo esté rellenado. Se utiliza para los elementos de Nombre, Edad, Presupuesto, Ciudad, Barrio y Aficiones. Un ejemplo es:

<asp:RequiredFieldValidator ID="rfv1" runat="server" ControlToValidate="tb1" ErrorMessage="No ha Introducido ningún nombre" SetFocusOnError="True" ></asp:RequiredFieldValidator>*

A este elemento, se le da un valor ID, se indica que será ejecutado en el servidor, que el control a validar es el Textbox llamado tb1, que muestre en caso de error el mensaje "no ha introducido ningún nombre", y que el foco de la navegación se centre en ese error. Por último, se añade un asterisco para ayudar al usuario a identificar el error.

RangeValidator: elemento a traves del cual se asegura que el valor representado por un control se encuentra dentro de un rango determinado. Se utiliza para Edad y Presupuesto. Un ejemplo es:

<asp:RangeValidator ID="rv10" runat="server" ControlToValidate="tb2" ErrorMessage="Introduzca un valor numerico entero" MaximumValue="150" MinimumValue="0" Type="Integer" Display="Dynamic" SetFocusOnError="True"></asp:RangeValidator>*

A este elemento se le da un valor ID, se identifica que será ejecutado en el servidor, que el control a validar es el Textbox llamado tb2, que en caso de error muestre el mensaje "Introduzca un valor numérico entero", que el tipo de datos sea un Integer, cuyo valor máximo es 150 y el mínimo 0 y que el foco de la navegación se centre en ese error. Por último, se añade un asterisco para ayudar al usuario a identificar el error.

CustomValidator: es un elemento que ayuda a proporcionar una ayuda para especificar funciones JavaScript propias de validación entre el servidor y el cliente. Se utiliza para la validacion del campo de Ciudad y Barrio. Un ejemplo es:

<asp:CustomValidator ID="CustomValidator1" runat="server" ClientValidationFunction="ValidarListaSeleccionada" ControlToValidate="ListBox1" ErrorMessage="Debe Seleccionar una Ciudad" ValidateEmptyText="True"></asp:CustomValidator>*

En este elemento, se le da un valor ID, se identifica que será ejecutado en el servidor, que el elemento a validar es el ListBox1 y que en caso de error muestre le mensaje "Debe Seleccionar una Ciudad". Además se le indica la función JavaScript a la que llamará para validar dicho elemento, que en este caso es ValidadListaSeleccionada.

La función JavaScript es:

```
<script type="text/javascript">
  function ValidarListaSeleccionada(source, arguments) {
    var list = document.getElementById("ListBox1");
    if (list != null) {
      var listValue = list.value;
      if (listValue == "0")
        arguments.IsValid = false;
        else
          arguments.IsValid = true;
    }
  }
</script>
```

Dónde busca el elemento a validar, en este caso el Listbox1, y comprueba que si el valor dado es 0 (es el primer elemento de la lista y no es ninguna ciudad, por lo que no es válido) devuelve FALSE y si no, devuelve TRUE. Esta función va con las etiquetas <script> de HTML.

ValidationSummary: es un elemento que muestra un resumen de todos los errores de validación de la página. La página principal tiene un elemento ValidationSummary que mostrará el resumen al final de la misma. El codigo es:

```
<asp:ValidationSummary ID="sm1" runat="server" DisplayMode="List"
    Font-Overline="False" Font-Size="Medium"
    HeaderText="LISTA ERRORES, por favor revise los siguientes errores:"
    ShowMessageBox="True" />
```

A este elemento se le da un ID, se le indica que será ejecutado en el lado del servidor, que muestre el mensaje en caso de error “LISTA ERRORES, por favor revise los siguientes errores:", el tamaño de letra y que el modo de mostrar la información sea en forma de Lista.

4.2.1 Funcionamiento Básico

Una vez validado el formulario con los datos introducidos por el usuario, se debe realizar la recomendación con las funciones CLIPS que se ha implementado anteriormente.

En la siguiente Ilustración se puede observar el funcionamiento.

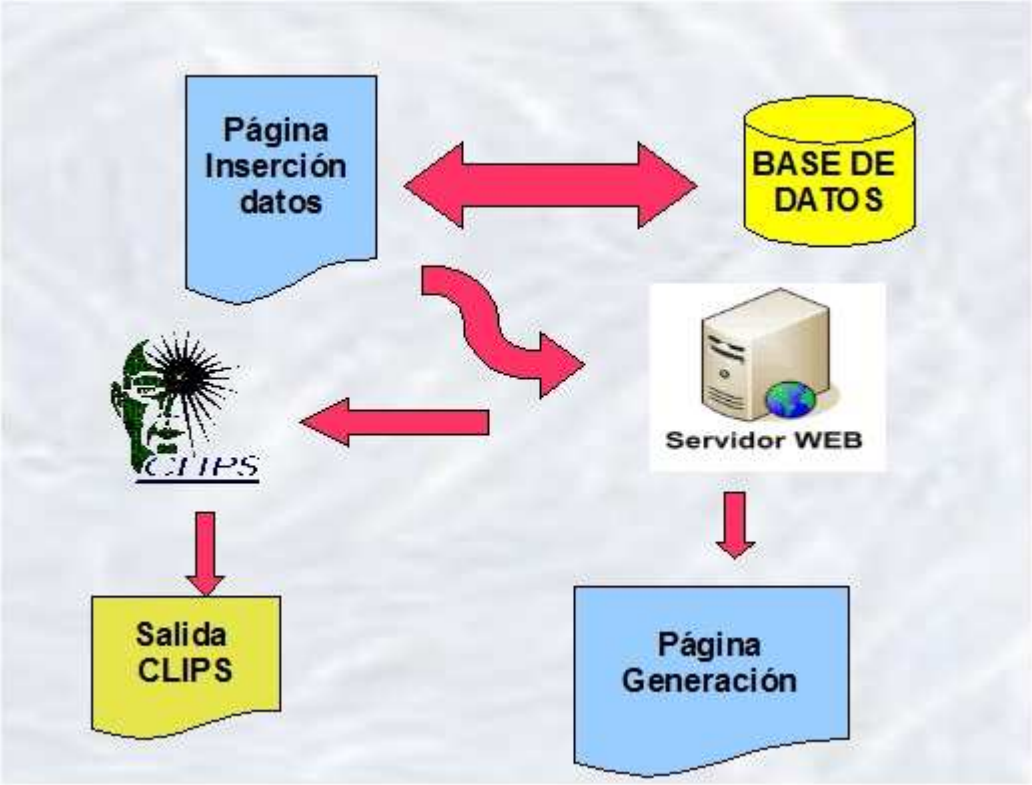


Ilustración 14: Funcionamiento Aplicación

La página de la inserción de datos por parte del usuario, se introducen los diferentes datos y preferencias, interactuando con la base de datos para las ciudades y los barrios correspondientes. Una vez validado se manda la petición al servidor.

En el servidor, a través del código C# que acompaña a la página .aspx, se crean las diferentes instancias de la persona con sus datos personales y aficiones, las instancias de preferencias horarias y por último las instancias correspondientes a la ciudad elegida, realizando una búsqueda en la base de datos y transformando los resultados para crear las instancias de las actividades de la ciudad.

Una vez realizadas las transformaciones, se hace el llamamiento a las funciones de CLIPS, a través de la librería CLIPSNET.dll, librería de C# que simula CLIPS sin tener que ejecutar la versión oficial. Estas llamadas se realizan a través del siguiente código:

ClipsNET.ClipsEngine clp = new ClipsNET.ClipsEngine();

```
clp.Reset();
clp.Load("clases.clp");
clp.Load("reglas.clp");
clp.Load("instancias.clp");
clp.Load("persona.clp");
clp.Reset();
clp.Run();
```

Primero se crea la instancia de ClipsEngine que es el motor de CLIPS en la librería. Posteriormente se hace un reset para inicializar la base de hechos, se cargan las clases, reglas, instancias de las actividades y por último la instancia de la persona. Una vez cargados todos estos ficheros, se vuelve a hacer un reset en la base de hechos e instancias para que se carguen. Por último, se le indica que haga Run para que ejecute las reglas. El código CLIPS ejecutará las diferentes reglas hasta acabar con la recomendación.

Finalmente, antes de redirigir a la página con las recomendaciones, es necesario utilizar algún método para poder mostrar al usuario su generación. Se utiliza una cookie, en la que se guardará el nombre del fichero con la generación de la recomendación y también la ciudad, para luego poder cargar el mapa de dicha ciudad. Un ejemplo de código es:

```
HttpCookie cok = new HttpCookie("gen",fecha);
 cok.Expires=DateTime.Now.AddDays(1);
 Response.Cookies.Add(cok);
```

Se genera una nueva instancia de la clase HttpCookie, llamada gen y se le pasa la fecha, que es valor identificativo. Se indica que expira en un día y por último se envía el evento Response añadiendo la cookie generada.

En la página Generacion.aspx (la página con las recomendaciones), se utiliza el evento de ASP.NET Page_Load para cargar los diferentes elementos de la página. Se lee primero la cookie, esto se hace con la siguiente instrucción.

```
HttpCookie coger = Request.Cookies.Get("gen");
```

A través del elemento Request, se coge la cookie llamada gen, que es la que tiene el id fecha para poder cargar la recomendación realizada al usuario. Se lee el archivo, y

se van cargando en la página las recomendaciones realizadas en el calendario. Al mismo tiempo, se van creando las tuplas en la base de datos con las recomendaciones realizadas. En cada botón de las recomendaciones, se guarda información sobre cada una de las tuplas que corresponde. Por último, se carga el mapa de Google Maps con la ciudad elegida.

Cuando el usuario pulse sobre uno de los botones de la recomendación, le aparecerá la información sobre la actividad recomendada, haciendo una petición al servidor para mostrar dicha información. El servidor accede a la base de datos y muestra la información en la página, actualizando además el mapa, para centrarse en la dirección de dicha actividad a través de las diferentes funciones que ofrece la API de Google Maps. Para ello, realiza las siguientes operaciones:

1) Configuración de la conexión a la base de datos.

```
ConnectionStringSettings conexion = ConfigurationManager.ConnectionStrings ["prueba1ConnectionString4"];
string cadenacon = conexion.ConnectionString;
SqlConnection conexion2 = new SqlConnection(cadenacon);
```

2) Preparación búsqueda

```
string busqueda = "select nombre,direccion,tipo,web from ACTIVIDAD where id=" + id;
SqlCommand ins2 = new SqlCommand(busqueda, conexion2);
conexion2.Open();
SqlDataReader rd2 = ins2.ExecuteReader();
```

3) Realización búsqueda

```
rd2.Read();
```

4) Mostrar resultados

```
tb1.Text = rd2[0].ToString();
tb2.Text = rd2[1].ToString();
tb3.Text = rd2[2].ToString();
tb4.Text = rd2[4].ToString();
tb5.Text = rd2[3].ToString();
```

5) Cerrar conexiones

```
rd2.Close();
conexion2.Close();
```

A la API de Google Maps se le pasa el valor de dirección para que lo cargue en el mapa, centrándose en dicha dirección.

4.2.2 RECOMENDACIONES

Para el apartado de Recomendaciones de otros usuarios, se realiza una búsqueda en la tabla RECOMENDACIONES. Se realiza una búsqueda de las recomendaciones hechas a partir de la actividad mostrada, para mostrar los 3 resultados que más veces han sido recomendados a partir de la experiencia de los usuarios. Con los resultados obtenidos, se muestra una foto identificativa y el nombre de la recomendación ofrecida.

Si el usuario pulsa sobre la recomendación, se muestra la información sobre dicha actividad del mismo modo que se muestra la información de una recomendación normal. Además se añade la opción al usuario para que si desea, pueda cambiar la recomendación realizada por el sistema por la que está viendo.

Si ocurre esto, se actualizará la recomendación, cambiando la recomendación establecida por el sistema (en el calendario) por la que el usuario ha elegido. Para ello se siguen los siguientes pasos:

1) Se crea la conexión (igual que el paso 1 de la anterior explicación)

2) Se prepara la búsqueda, también es igual que la anterior, pero la cadena a ejecutar es distinta:

   string cadenainsert = "INSERT INTO GENERACION (id, dia, franja, fecha, actividad, precio, actual) VALUES (";
   cadenainsert += fecha2 + ", " + dia + ", " + franja + ", " + fecha + ", " + identificador + ", "+ precio+",1)";

3) Se ejecuta la sentencia:

   SqlCommand ins2 = new SqlCommand(cadenainsert, conexion2);
   ins2.ExecuteNonQuery();

4) Se cierran las conexiones:

```
ins2.Dispose();
ins2= null;
conexion2.Close();
```

Se cambia en el botón correspondiente la información referente a la actividad que recomienda para esa franja y día determinada.

Una vez realizado el cambio, se debe actualizar los datos de la tabla RECOMENDACIONES, modificando y aumentando el contador de las recomendaciones realizadas. Para ello se crea una función llamada actualizar recomendación a través de la cual se actualizará el contador.

1) El encabezado es:

   private void actualizarrecomendacion(int id, int id2,int valor)

2) Posteriormente, se crea una conexión a la base de datos del mismo modo que se ha realizado en anteriores procedimientos.

3) Se realiza la siguiente búsqueda para obtener el dato del contador de recomendaciones llamado total.

```
string cadenainsert = "select total FROM RECOMENDACION where";
 cadenainsert += "(actividad2=" + id + ")AND(actividad1=" + id2+")";
```

4) Una vez obtenido este dato, se le añade un valor determinado. En caso de que el usuario pulse para ver la información de dicha actividad, se le añade 1. En caso de que el usuario cambie la recomendación efectuada por el sistema, por la actividad que está viendo la información, se añade 5.

5) Finalmente, se realiza la actualización en la base de datos:

```
 string cadenainsert2 = "UPDATE RECOMENDACION SET total=" + total + "
WHERE (actividad2=" + id+")AND(actividad1=" + id2+")";
 SqlCommand ins = new SqlCommand(cadenainsert2, conexion2);
 conexion2.Open();
 ins.ExecuteNonQuery();
 ins.Dispose();
 ins = null;
 conexion2.Close();
```

4.2.3 Generación Informe PDF

Una vez realizada la recomendación y usuario haya podido ver y cambiar las distintas actividades asignadas, puede descargarse un archivo PDF con las recomendaciones realizadas según sus preferencias. Para ello, se utiliza la librería iTextSharp que permite crear diferentes documentos, entre ellos un documento PDF.

Se utiliza una cookie en la que se guarda el nombre del usuario para realizar la recomendación, se crea la instancia del documento y se abre el documento.

```
HttpCookie coger2 = Request.Cookies.Get("nombre");
string nombre = "Recomendacion para "+coger2.Value+".pdf";
Document document = new Document();
PdfWriter.GetInstance(document,newFileStream(nombre,FileMode.OpenOrCreate);
document.Open();
```

Para añadir un párrafo, es tan sencillo como añadir la siguiente sentencia:

```
document.Add(new Paragraph("Recomendacion"));
```

A continuación, se recorren todas las actividades de la generación y se imprime la información necesaria, como es el día la franja, la actividad, la dirección, la zona, el medio de transporte y el precio de la recomendación. Al final del documento, se indica al usuario el total de las actividades. Por último, se cierra el archivo con la siguiente sentencia:

```
document.Close();
```

Una vez generado el documento PDF, se crea una nueva cookie en la que guardar el nombre del archivo y se redirecciona a la página descargar.aspx.

```
HttpCookie archivo = new HttpCookie("archivo", ruta2);
archivo.Expires = DateTime.Now.AddDays(1);
 Response.Cookies.Add(archivo);
 Response.Redirect("descargar.aspx");
```

En la página descargar.aspx, se lee la cookie generada anteriormente con el nombre del archivo.

```
HttpCookie coger = Request.Cookies.Get("archivo");
string ruta = coger.Value.ToString();
```

Por último, se muestra el archivo con las sentencias Response, estas sentencias son:

```
Response.Clear();
string filename = "attachment; filename=" + ruta;
Response.ContentType = "application/pdf";
Response.AddHeader("Content-Type", "application/pdf");
Response.AddHeader("Content–Disposition", filename);
Response.TransmitFile(ruta);
Response.End();
```

El usuario puede ver en su navegador la recomendación personalizada, pudiendo guardar dicho documento. Un ejemplo de la generación del documento PDF es:

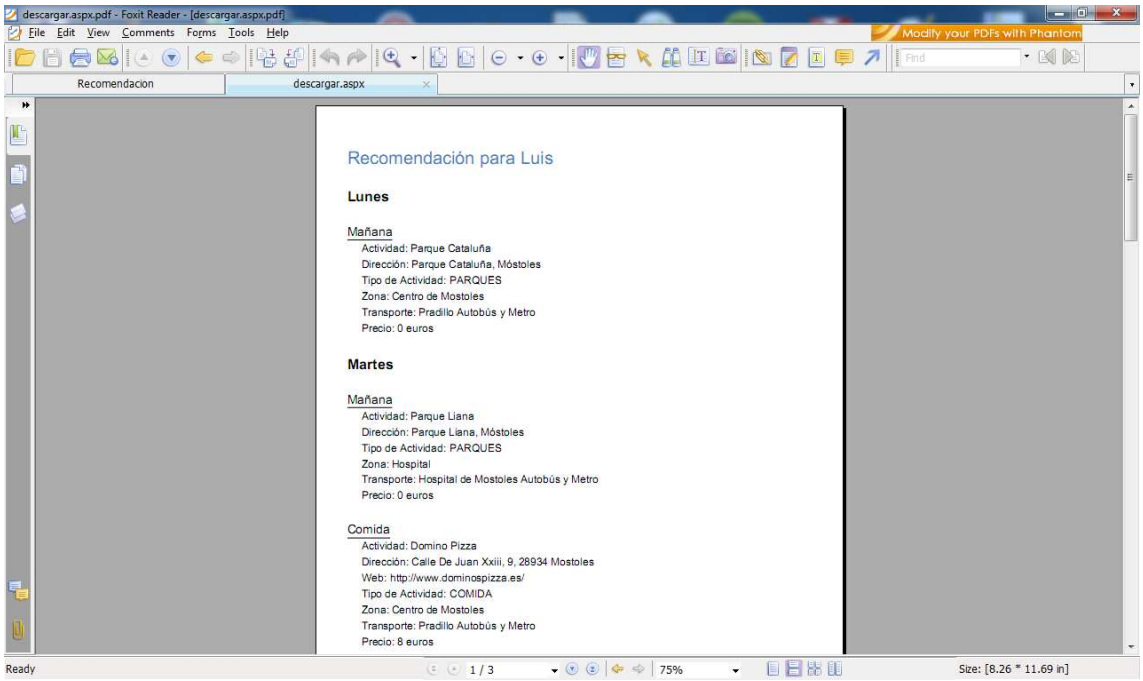


Ilustración 15: Ejemplo Generación Documento PDF

4.3 PRUEBAS

En esta sección se realizan diferentes pruebas para comprobar el correcto funcionamiento de la aplicación, poniendo especial cuidado en la robustez del sistema y la coherencia de los diferentes datos dados por el usuario y su resultado.

4.3.1 VALIDACIÓN FORMULARIO

Para realizar la recomendación del usuario, es necesario validar el formulario en el que el usuario debe introducir sus datos y sus preferencias, así como la disponibilidad horaria.

Datos Personales:

Ilustración 16: Inserción Datos Personales

Prueba 1: Dejar todos los campos vacíos. El sistema informa de qué campos se han dejado vacíos y pone el foco de la navegación sobre el primer error.

Prueba 2: Introducir letras en campos edad y presupuesto. El sistema informa de que los campos edad y presupuesto presentan valores no válidos y se pone el foco de la navegación sobre el campo edad.

Prueba 3: Introducción de datos correctos. El sistema valida correctamente los diferentes datos.

Preferencias Usuario

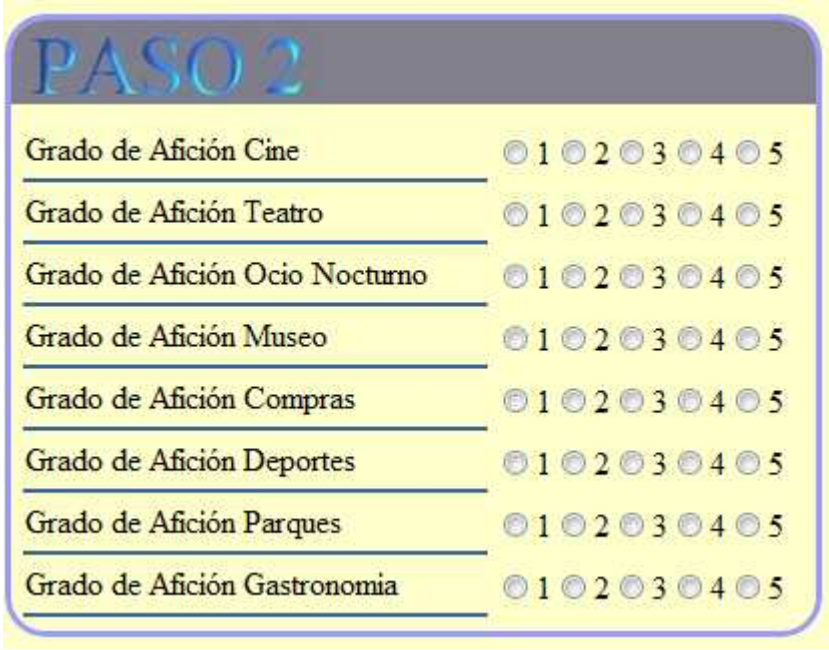


Ilustración 17: Elección Grados de Afición

Prueba 1: No seleccionar ningún campo. El sistema informa que no se ha seleccionado ningún valor para los diferentes grados de afición, centrando el foco de la navegación en el primer campo erróneo, que en este caso es Cine.

Prueba 2: Dejar un campo (por ejemplo Museo) sin seleccionar ningún valor de preferencias. El sistema informa de que el campo Grado de Afición de Museos no ha sido seleccionado ningún valor, centrando el foco de la navegación en dicho campo.

Prueba 3: Seleccionar correctamente las diferentes preferencias. El sistema valida correctamente los campos.

Disponibilidad Horaria

Prueba 1: Dejar todos los campos vacíos. El sistema informa de que no ha sido seleccionada ninguna disponibilidad horaria (día y franja), centrando el foco de la navegación en el calendario.

Prueba 2: Rellenar la disponibilidad. El sistema valida correctamente el formulario.

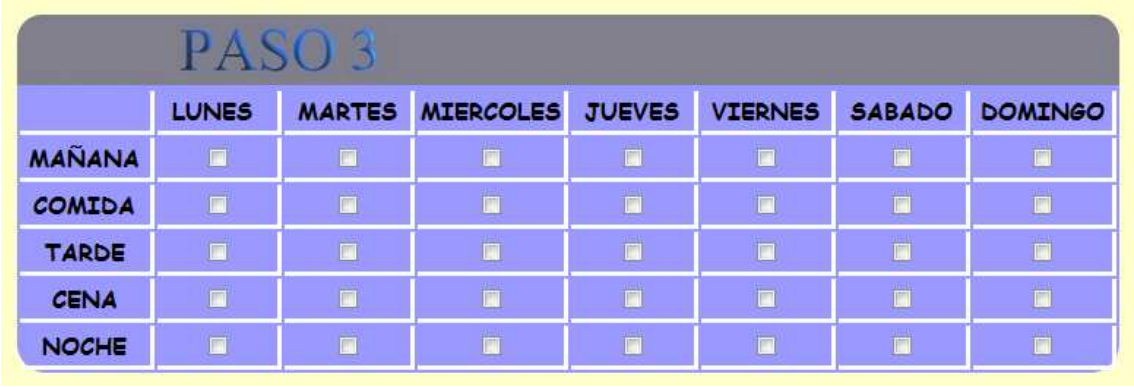


Ilustración 18: Disponibilidad Horaria

4.3.2 PRUEBAS DE LA APLICACIÓN

Dentro de esta sección, se prueban distintas recomendaciones con diferentes valores de los datos de usuario (edad y presupuesto), discapacidades, grados de afición y disponibilidad horaria.

Presupuesto bajo

Se prueba con un presupuesto de 0 euros, y marcando grado de afición de todas las actividad a nivel 5 y por último, marcando todas los días y franjas.

Entrada datos

Edad	30	Grado Cine	5
Presupuesto	0	Grado Teatro	5
Discapacidad visual	No	Grado Restauración	5
Discapacidad auditiva	No	Grado Ocio Nocturno	5
Discapacidad motriz	No	Grado Museo	5
Disponibilidad Horaria TOTAL		Grado Compras	5
		Grado Deportes	5
		Grado Parques	5

SALIDA

La salida del sistema muestra 4 recomendaciones, que son las actividades de parques, cuyo coste es de 0 euros, como es lógico, no ha recomendado ninguna actividad más ya que todas tienen un coste y el presupuesto es de 0 euros.

Edad inferior a 18

Se prueba con una edad de 16 años, y marcando a 5 el grado de afición por el Ocio Nocturno y las demás actividades a 1, y por último en la disponibilidad horaria solo marcamos las noches, ya que son las franjas donde solo está disponible el ocio nocturno y que dispone de prohibición de edad para menores de 18 años.

Entrada Datos

Edad	16	Grado Cine	1
Presupuesto	2220	Grado Teatro	1
Discapacidad visual	No	Grado Restauración	1
Discapacidad auditiva	No	Grado Ocio Nocturno	5
Discapacidad motriz	No	Grado Museo	1
Disponibilidad Horaria Solo Noches		Grado Compras	1
		Grado Deportes	1
		Grado Parques	1

SALIDA

El sistema no devuelve ninguna recomendación, ya que el usuario no puede participar en ninguna actividad nocturna debido su edad

Discapacidad motriz

Se prueba indicando que se padece una discapacidad motriz y indica grado de afición máximo en actividades que sabemos que no está adaptado a la discapacidad motriz. La disponibilidad horaria se rellena completamente. Por ejemplo, sabemos que Teatro Caleidoscopio, Teatro Joan Miró, Polideportivo de Villafontana y el estadio de fútbol no están adaptados para discapacidad motriz, por lo tanto pondremos grado de afición máximo a teatro y deportes.

Entrada Datos

Edad	30	Grado Cine	1
Presupuesto	200	Grado Teatro	5
Discapacidad visual	No	Grado Restauración	1
Discapacidad auditiva	No	Grado Ocio Nocturno	1
Discapacidad motriz	SI	Grado Museo	1
Disponibilidad Horaria TOTAL		Grado Compras	1
		Grado Deportes	5
		Grado Parques	1

SALIDA

El sistema recomienda actividades de teatro y deportes, pero no las actividades señaladas anteriormente que no disponen de accesibilidad para personas con discapacidad motriz.

Discapacidad auditiva

Se prueba indicando que se padece una discapacidad auditiva y indica grado de afición máximo en actividades que sabemos que no está adaptado a la discapacidad auditiva. La disponibilidad horaria se rellena completamente. Por ejemplo, sabemos que ningún teatro ni actividad deportiva están adaptados a la discapacidad auditiva, por lo tanto pondremos grado de afición máximo a teatro y deportes.

Entrada Datos

Edad	30	Grado Cine	1
Presupuesto	200	Grado Teatro	5
Discapacidad visual	No	Grado Restauración	1
Discapacidad auditiva	SI	Grado Ocio Nocturno	1
Discapacidad motriz	No	Grado Museo	1
Disponibilidad Horaria TOTAL		Grado Compras	1
		Grado Deportes	5
		Grado Parques	1

SALIDA

El sistema no recomienda ninguna actividad ya que ninguna actividad de deportes y teatro está adaptada para personas con discapacidad auditiva.

Discapacidad visual

Se prueba indicando que se padece una discapacidad visual y indica grado de afición máximo en actividades que sabemos que no está adaptado a la discapacidad visual. La disponibilidad horaria se rellena completamente. Por ejemplo, sabemos que ningún teatro ni actividad deportiva están adaptados a la discapacidad visual, por lo tanto pondremos grado de afición máximo a teatro y deportes.

Entrada Datos

Edad	30	Grado Cine	1
Presupuesto	200	Grado Teatro	5
Discapacidad visual	Si	Grado Restauración	1
Discapacidad auditiva	No	Grado Ocio Nocturno	1
Discapacidad motriz	No	Grado Museo	1
Disponibilidad Horaria TOTAL		Grado Compras	1
		Grado Deportes	5
		Grado Parques	1

SALIDA

El sistema no recomienda ninguna actividad ya que ninguna actividad de deportes y teatro está adaptada para personas con discapacidad visual.

Grados de afición

Se prueban diferentes valores de los grados de afición para comprobar la correcta asignación de las diferentes actividades.

Entrada Datos

Edad	30	Grado Cine	3
Presupuesto	200	Grado Teatro	2
Discapacidad visual	Si	Grado Restauración	4
Discapacidad auditiva	No	Grado Ocio Nocturno	4
Discapacidad motriz	No	Grado Museo	4
Disponibilidad Horaria Lunes, Martes y Miércoles completos		Grado Compras	3
		Grado Deportes	2
		Grado Parques	1

SALIDA

El sistema muestra diferentes recomendaciones, se comprueba que las recomendaciones con más valor son las que realmente han sido las que han sido seleccionadas, y que no se sobrepase el presupuesto asignado.

Mirar una recomendación

Cuando un usuario selecciona una asignación realizada por el usuario en un determinado día y franja, se le muestra toda la información de esa actividad (nombre, dirección, precio, web, transporte, etc.) y además en la parte inferior, otras recomendaciones basadas en la experiencia de navegación de otros usuarios, mostrando una imagen representativa y el nombre. Si el usuario pincha sobre la imagen, se le muestra toda la información sobre esa actividad y se ofrece la posibilidad de cambiar la asignación del sistema por la que le ofrece la recomendación de otros usuarios. Se prueba que las recomendaciones ofrecidas son coherentes con la actividad asignada por el sistema.

Cambiar recomendación

Si el usuario decide realizar el cambio de la recomendación, el sistema debe cambiar la asignación efectuada por dicho sistema por la que el usuario ha elegido cambiar. Se comprueba que efectivamente, realiza el cambio correctamente.

Generación PDF antes y después del cambio

Se genera una recomendación, en la se asigna por ejemplo al usuario ir al polideportivo Villafontana el martes por la tarde. Se genere el PDF con las recomendaciones y se guarda. A continuación, se cambia dicha asignación por una de las recomendaciones efectuadas, por ejemplo, el polideportivo El Soto. Se vuelve a generar el PDF con las recomendaciones, y se observa que el cambio se ha producido también en la recomendación, por lo que el sistema realiza el cambio correctamente.

4.4 VALIDACIÓN HMTL Y CSS

Una vez realizada la aplicación, es importante validar el código de la página web para que cumpla con los estándares de la *World Wide Web Consortium* (W3C), y que es la encargada de crear los estándares de la Web. A través de la página http://validator.w3.org/ se valida el código HTML de la página.

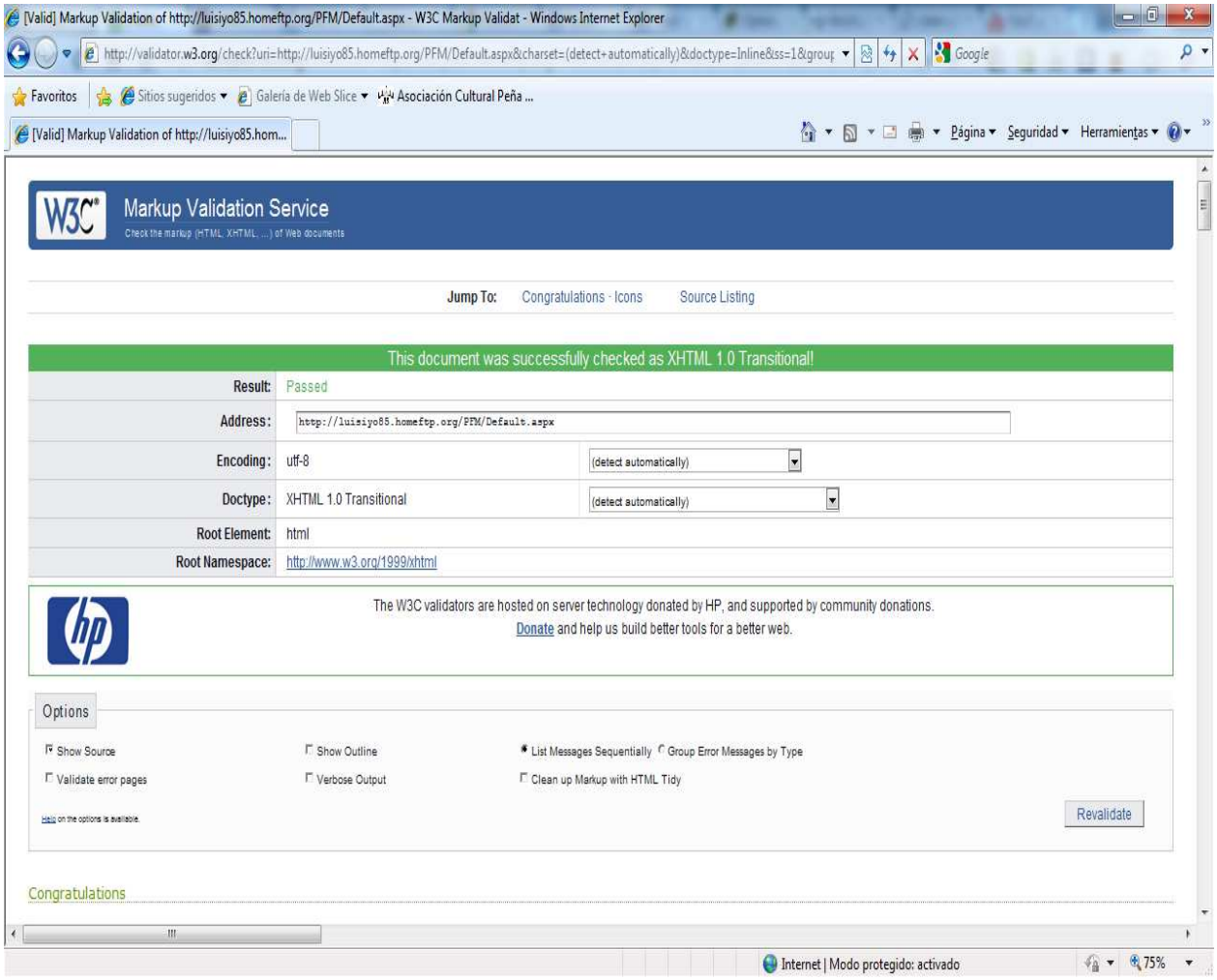


Ilustración 19: Validación Código HTML

Como se puede comprobar, la validación del código HTML ha sido correcta. A continuación se debe validar el estilo de la página. Esta operación se realiza a través de la página http://jigsaw.w3.org/css-validator/ .

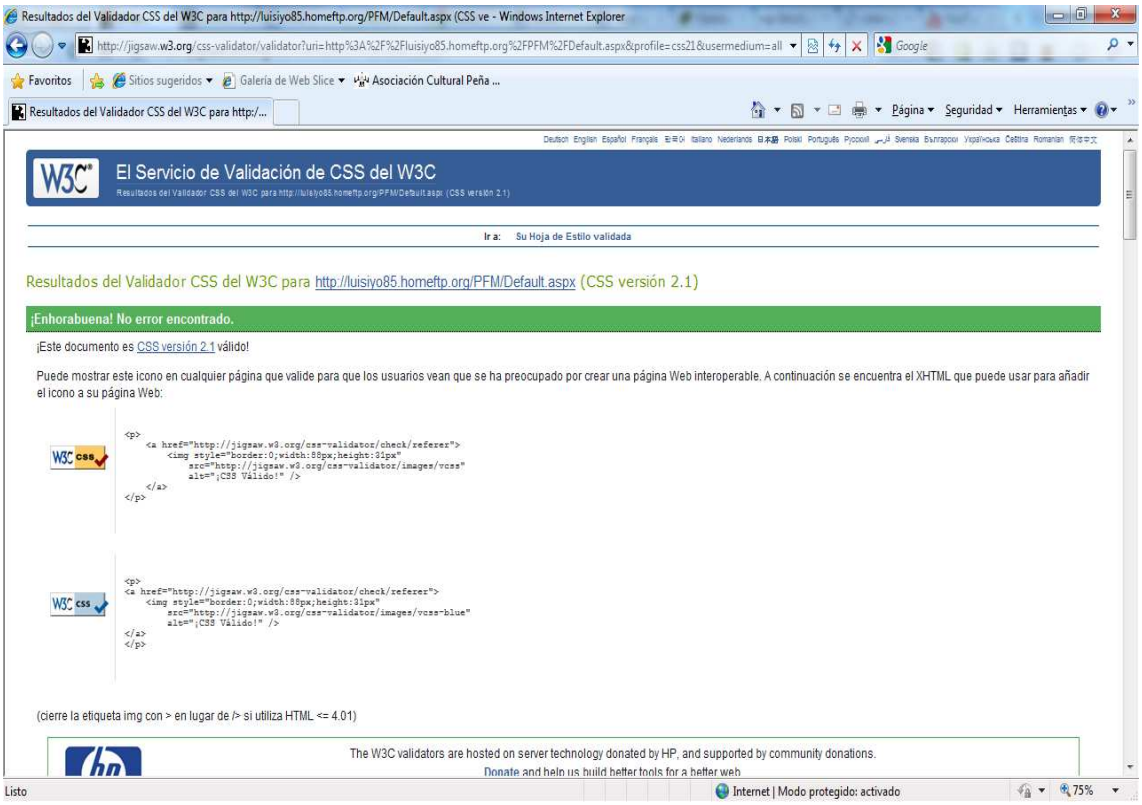


Ilustración 20: Validación Hoja de estilo

La validación de la hoja de estilos ha sido correcta. Como ambas validaciones han sido correctas, se puede asegurar que la página Web del proyecto cumple con los diferentes estándares para la Web publicados por la W3C.

4.5 VISION EN OTROS NAVEGADORES

Debido a la gran cantidad de navegadores existentes y que ninguno cumple estrictamente los estándares HTML de la *World Wide Web Consortium* (W3C), es preciso realizar pruebas en distintos navegadores para comprobar que los distintos elementos no aparecen descolocados o mal presentados, en definitiva, que la disposición de la página sea la misma sea cual sea el navegador desde el que se accede a la aplicación.

4.5.1 INTERNET EXPLORER 8

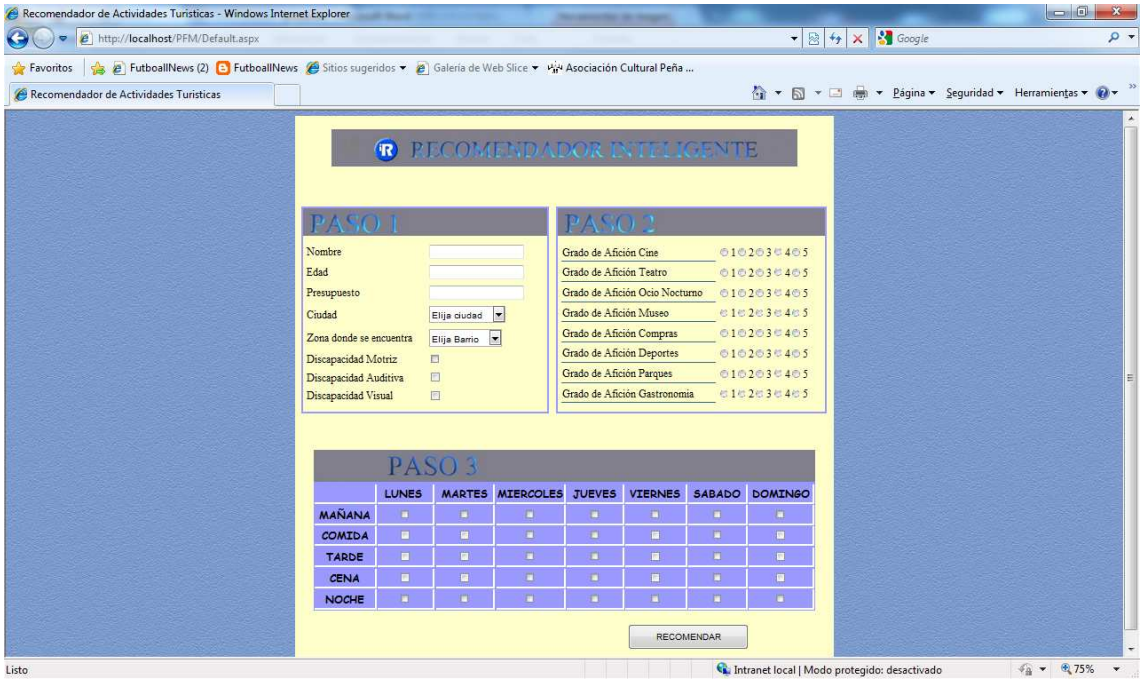


Ilustración 21: Internet Explorer 8 Página 1

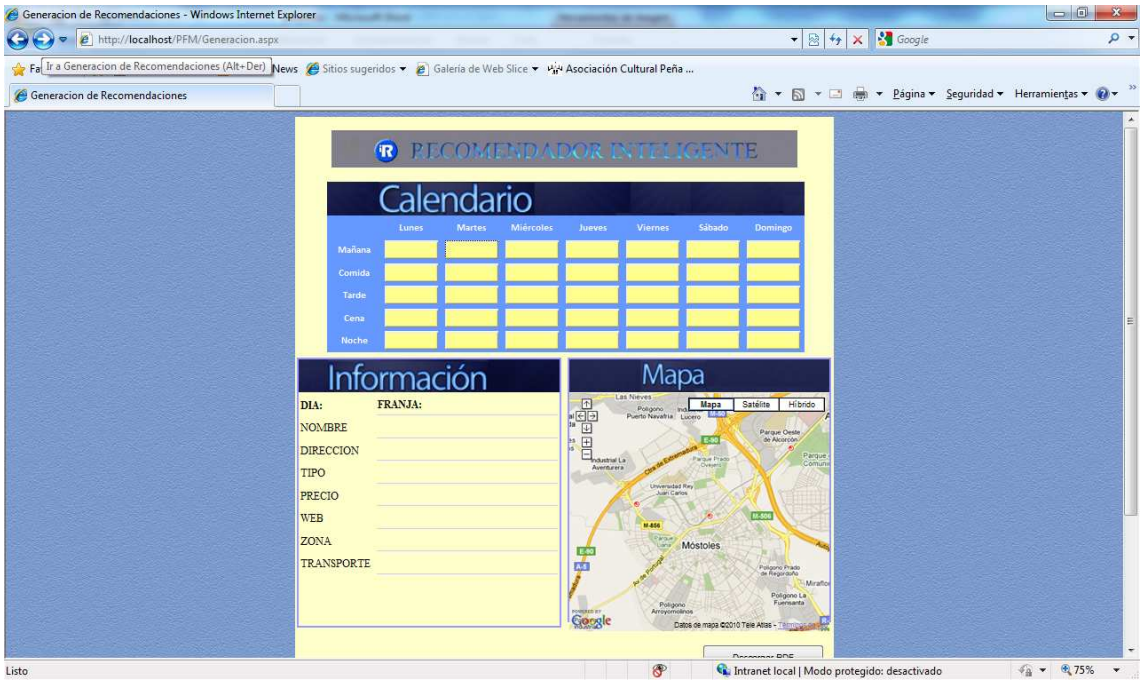


Ilustración 22: Internet Explorer 8 Página 2

4.5.2 GOOGLE CHROME 5.0

Ilustración 23: Google Chrome Página 1

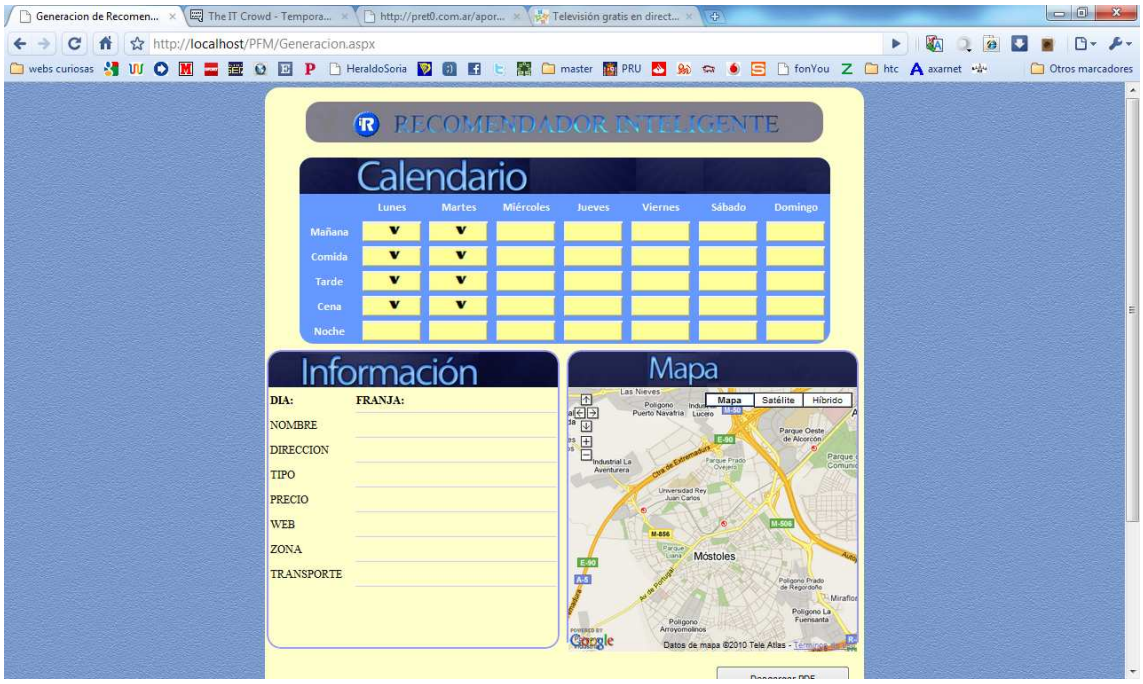


Ilustración 24: Google Chrome Página 2

4.5.3 MOZILLA FIREFOX (3.5.7)

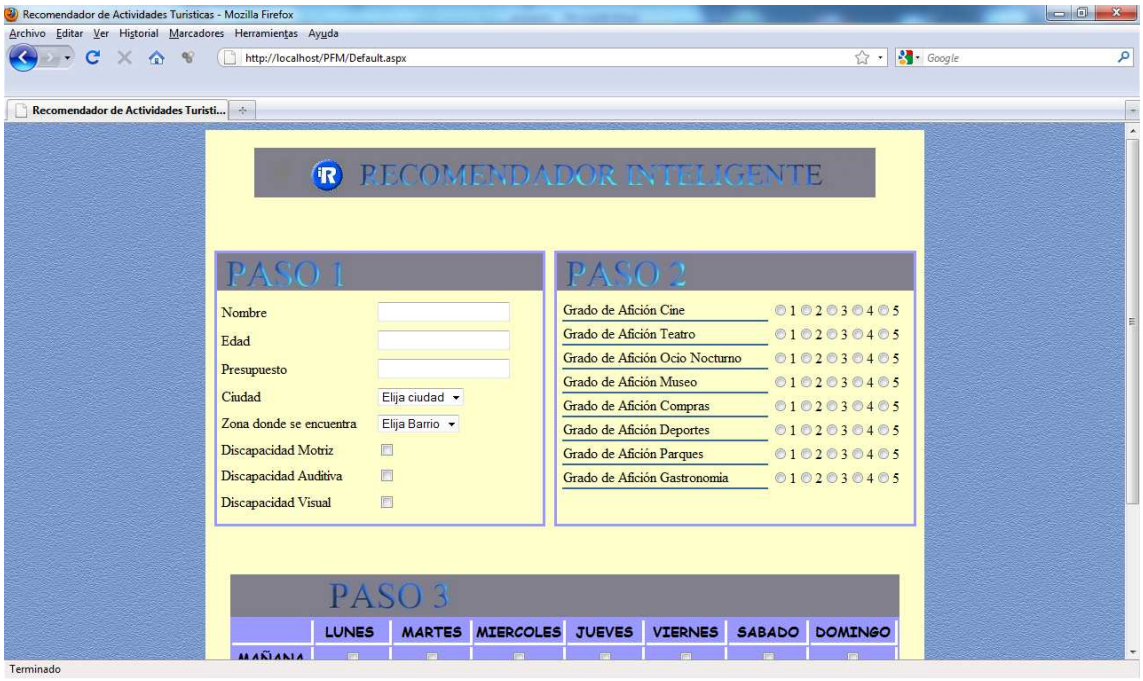


Ilustración 25: Firefox Página 1

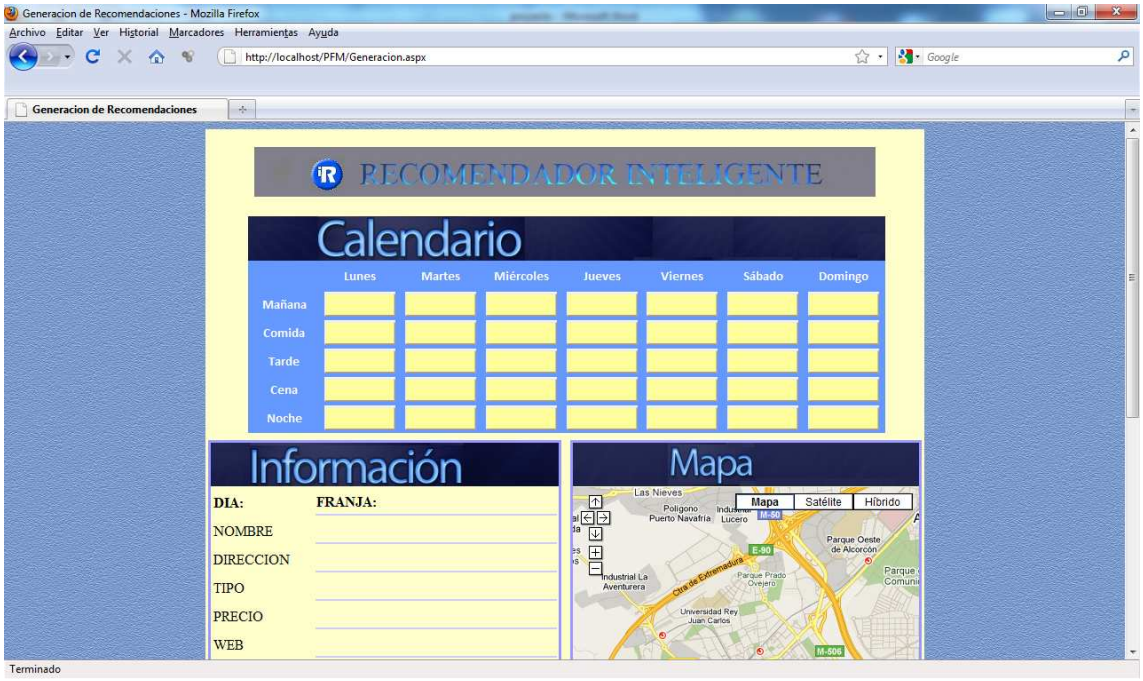


Ilustración 26: Firefox Página 2

4.5.4 OPERA (10.53)

Ilustración 27: Opera Página 1

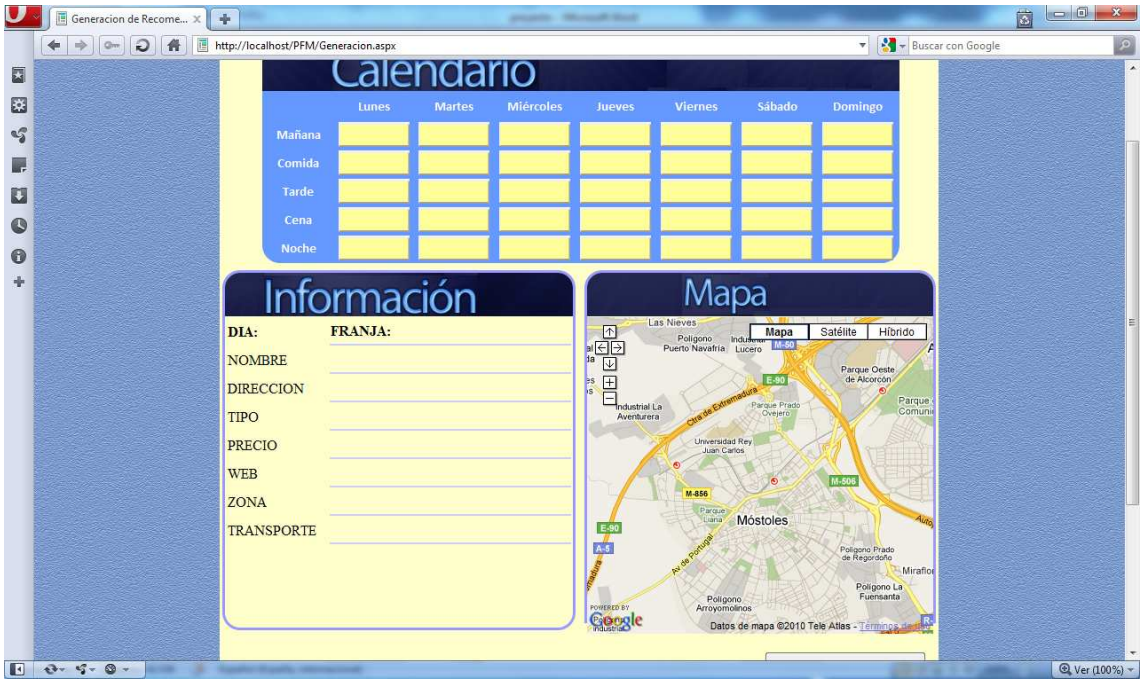


Ilustración 28: Opera Página 2

4.5.5 SAFARI 5.0

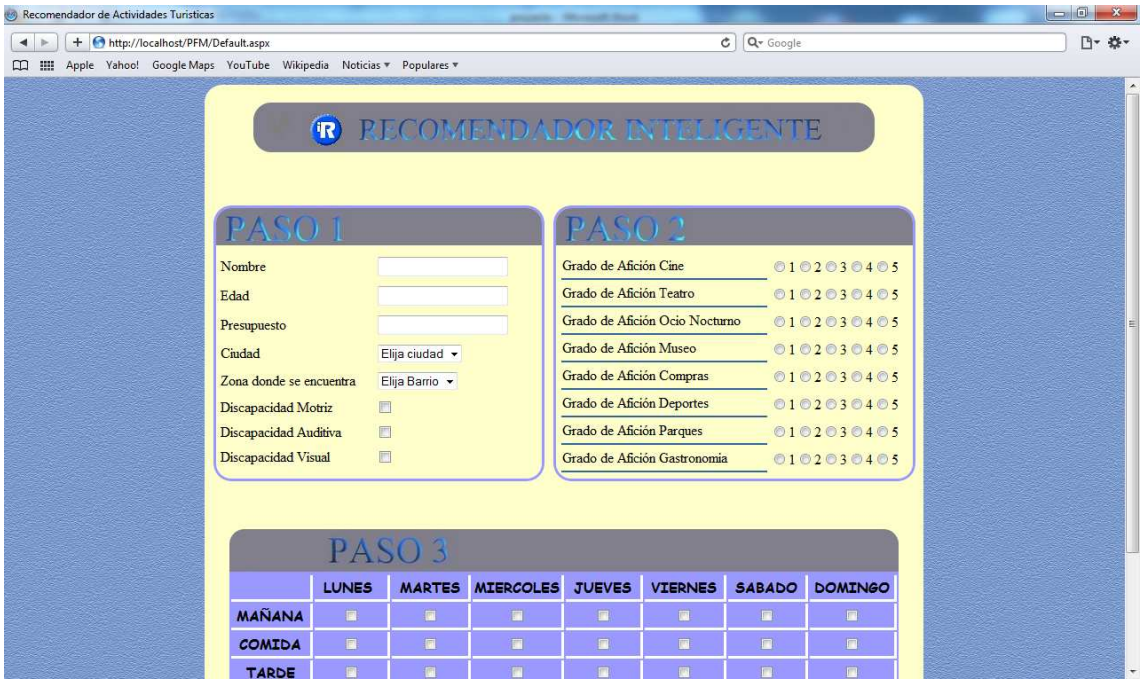


Ilustración 29: Safari Página 1

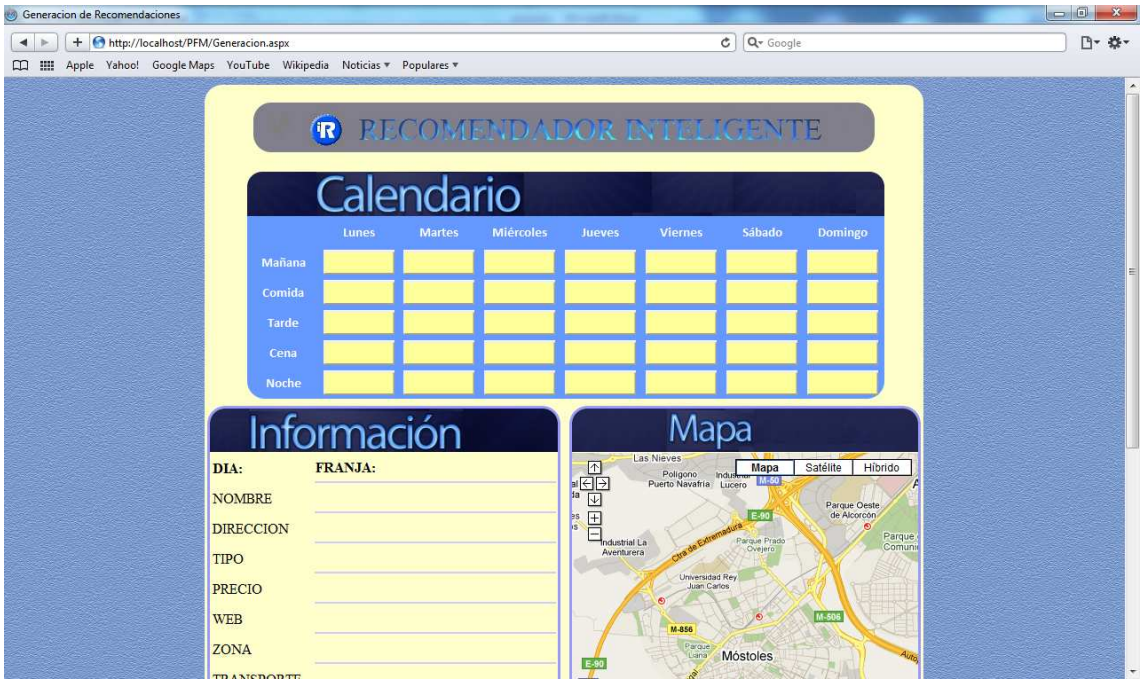


Ilustración 30: Safari Página 2

5. CONCLUSIONES Y TRABAJOS FUTUROS

En el desarrollo del proyecto, se han utilizado diferentes técnicas, lenguajes y sistemas para poder realizar una aplicación acorde con las necesidades de los diferentes usuarios, accesible a todos y que cumpla la misión de realizar recomendaciones personalizadas conforme a las preferencias y los datos personales proporcionados por el usuario.

La realización de una ontología facilita la implementación de la jerarquía de clases para la división de las diferentes actividades, y con la utilización del *software Protégé*, se ha simplificado más aún esta tarea, ya que a través de su interfaz se consigue realizar la ontología de una forma fácil y sencilla.

La utilización de la tecnología ASP.NET permite crear un sitio Web con todas las garantías, ya que la potencia y la facilidad de uso que aporta de esta tecnología hacen que el desarrollo de un sitio sea mas sencillo, además de poder realizar sitios webs dinámicos, pudiendo programar el comportamiento de los distintos elementos de la página, dando un poder similar al de una aplicación de escritorio. Además, la sencillez a la hora de interactuar con la base de datos facilita enormemente su uso, ya que se añade la potencia de estas a la aplicación Web.

Un punto fuerte de este proyecto han sido los sistemas de recomendación, donde a través de un sistema de filtrado colaborativo se realizan las recomendaciones a los usuarios en función de la actividad recomendada por el sistema.

Como trabajos futuros se pueden realizar diferentes mejoras en la accesibilidad del sitio Web para facilitar la navegación en este por todo tipo de usuarios, sea cual sea su capacidad física, intelectual y tecnológica.

Una opción muy recomendable es la de crear una página dentro del portal, donde se puedan incluir nuevas ciudades o actividades en la aplicación y hacerlo de una manera sencilla, sin tener que implementar ninguna consulta o inserción en la base de datos.

En lo que respecta el sistema de recomendación basado en filtrado colaborativo, una de las posibles mejorar puede ser la de mezclar los diferentes resultados con otros sistemas de recomendación como puede ser el de las sugerencias basadas en factores económicos o demográficos.

6. BIBLIOGRAFÍA

1. D. Goldberg, D. Nichols, B.M. Oki, y D. Terry, “Using collaborative filtering to weave an information tapestry,” 1992.
2. P. Resnick y H.R. Varian, “Recommender systems,” Communications of the ACM, vol. 40, 1997, pág. 57.
3. E.M. Housman y E.D. Kaskela, “State of the art in selective dissemination of information,” IEEE Transactions on Engineering Writing and Speech, vol. 13, 1970, págs. 78-83.
4. N. Manouselis y C. Costopoulou, “Analysis and Classification of Multi-Criteria Recommender Systems,” World Wide Web, vol. 10, 2007, págs. 415-441.
5. M. Balabanovic y Y. Shoham, “Content-based, collaborative recommendation,” Communications of the ACM, vol. 40, 1997, pág. 67.
6. R. Burke, “Knowledge-based recommender systems,” Encyclopedia of Library and Information Systems, vol. 69, 2000, págs. 175-186.
7. C. Basu, H. Hirsh, y W. Cohen, “Recommendation as classification: Using social and content-based information in recommendation”. Proc. of the National Conference on Artificial Intelligence, JOHN WILEY & SONS LTD, 1998, págs. 714-720.
8. REJA: un sistema de recomendación de restaurantes basado en técnicas difusas M.J. Barranco, L.G. Pérez, F. Mata, L. Martínez Universidad de Jaén.
9. Emilio J. Castellano, Luis Martinez, Manuel Barranco y Luis G. Pérez Cordón. Recomendación de perfiles académicos mediante algoritmos colaborativos basados en el expediente. IADIS International Conference, 2007.
10. Sieg, A., Mobasher, B., & Burke, R. Ontological User Profiles for Representing Context in Web Search. Proceedings of the Workshop on Web Personalization and Recommender Systems. At the ACM International Conference on Web Intelligence, Silicon Valley, CA, November 2007.
11. 2010 Stanford Center for Biomedical Informatics Research. Web page: protege.stanford.edu. Marzo 2010.
12. 2009 http://sourceforge.net/projects/clipsnet/ CLIPSNET.
13. 2009 SourceForge.net. Web page: clipsrules.sourceforge.net. Marzo 2010.
14. 2010 Microsoft Corporation. Microsoft asp.net Ajax. Web page: www.asp.net/ajax. Marzo 2010.
15. Cristina María Moraño Martínez. SiReTur. Sistema de Recomendaciones Lingüístico-Difuso aplicado al Turismo. Proyecto final de Ingeniería en Informática de Sistemas de la Universidad de Cordoba. Junio 2006.
16. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst., 22(1):5–53, 2004.
17. ASP.NET 3.5 Paso a Paso, Shepherd, George, Madrid : Anaya Multimedia, D.L. 2008.

18. Programación Web con Visual Studio y ASP.NET 2.0 [aprende a sacar todo el partido a la última versión de .NET], Alarcón Aguín, José Manuel, [s.l.] : Krasis Consulting, D.L. 2007, xvi, 235 p. : il. ; 25 cm, ISBN 9788493548902
19. Desarrollo de aplicaciones Web con ASP.NET 2.0, Martín Sierra, Antonio, Paracuellos de Jarama : Ra-Ma, D.L. 2007.
20. ASP.NET 2.0, Evjen, Bill, Madrid: Anaya Multimedia, D.L. 2006.
21. Inteligencia artificial un enfoque moderno, Russell, Stuart J., 2ª ed., última reimp. 2006, Madrid [etc.] : Pearson Prentice Hall, D.L. 2006.
22. SQL Server 2008 SQL, transact SQL diseño y creación de una base de datos, Gabillaud, Jérome, Cornellà de Llobregat : ENI, cop. 2009.
23. World Wide Web Consortium, W3C, http://www.w3.org/.
24. Google Maps API http://code.google.com/intl/es-ES/apis/maps/.
25. iTextSharp http://www.devjoker.com/contenidos/C/374/Generar-archivos-PDF-con-C.aspx.

APÉNDICE A: MANUAL DE USUARIO

Para la utilización correcta de la aplicación, se escribe este manual de usuario. Primeramente debe seguir estos 3 pasos.

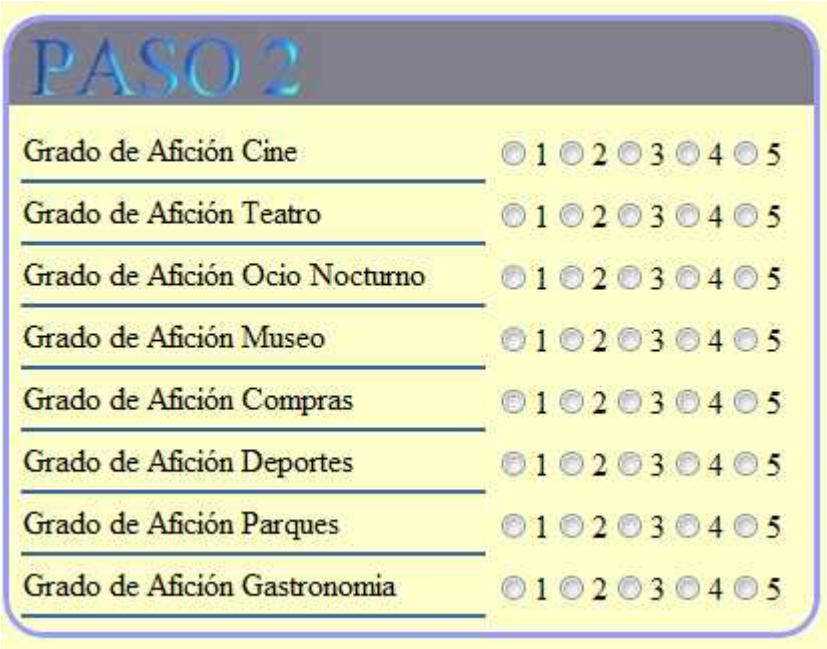
Paso 1: Introducción de Datos

Introduzca los datos necesarios para poder realizar una recomendación personalizada.

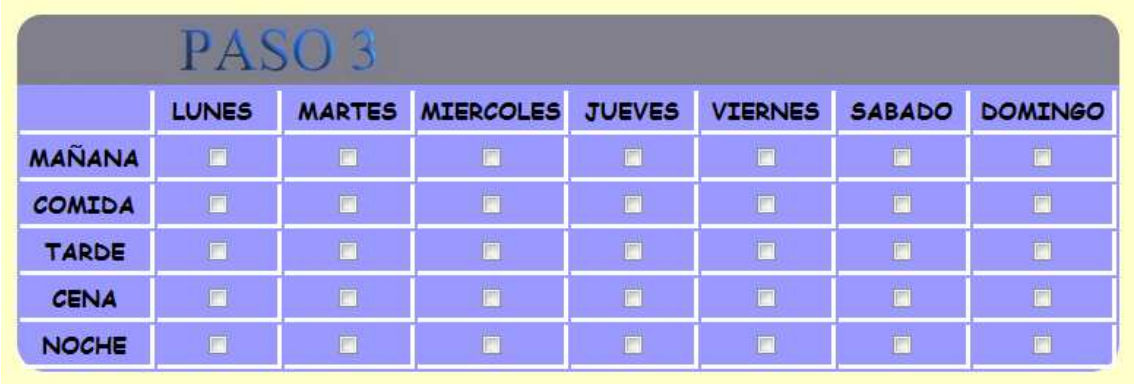
- Nombre de Usuario: para poder realizar la recomendación personalizadas.
- Edad: Hay determinadas actividades restringidas a ciertas edades.
- Presupuesto: El Dinero del de dispone para realizar las actividades.
- Ciudad y Zona: Ciudad para la que quiere buscar las recomendaciones.
- Discapacidades Motriz, Auditiva y Visual: Si tiene alguna o varias de estas discapacidades, usted puede indicarlo.

Paso 2: Aficiones

Indique de 1 a 5 su grado de afición para las siguientes actividades: Cine, Teatro, Ocio Nocturno, Museo, Compras, Deportes, Parques y Gastronomía.



Paso 3: Disponibilidad Horaria



En esta parte, usted debe elegir los días y horas que tiene disponible o que quiere recibir recomendaciones. Por último, Pulse el Botón RECOMENDAR para que sea realizada la recomendación.

A continuación, el sistema le redirigirá a la página con las actividades asignadas. En la siguiente página, se le muestra la información sobre las actividades que le han sido asignadas de acuerdo a sus preferencias, edad, presupuesto y disponibilidad horaria.

En la parte superior, le aparece un calendario, donde se le indica con el botón las asignaciones realizadas. En la parte inferior se le muestra la información de la actividad y el mapa de situación donde se encuentra la actividad.